UNIVERSIDAD CARLOS III DE MADRID

INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN
ESPECIALIDAD SONIDO E IMAGEN

PROYECTO FIN DE CARRERA

SISTEMA DE VISIÓN ESTEREOSCÓPICA AUTOCALIBRADO

Autor:
Jacobo Rouces González

Tutor:
Jose Jesús García Rueda

Diciembre 2009

RESUMEN

Este proyecto consiste en el diseño e implementación de un sistema de software que emplea visión estereoscópica a través de cámaras generales de bajo coste para ofrecer una interfaz tridimensional de entrada persona – máquina. Para que se pueda realizar un seguimiento en tiempo real, se admitirá el uso de un patrón físico visualmente distintivo. Además, el sistema debe tener la capacidad de autocalibrarse con mínima intervención humana, pudiendo así funcionar con diversidad tanto en las características como en la orientación relativa de las cámaras, sin que el usuario deba medir o resolver directamente los parámetros que definen cada situación.

ABSTRACT

This project consists in the design and implementation of a software system that makes use of stereoscopic vision through cheap off-the-shelf cameras, to offer a tridimensional Human Interface Device. So that the tracking may be done in real time, it will be acceptable the use of a visually distinctive physical pattern. Besides, the system must be able to self-calibrate with minimal human intervention, working with different kinds of cameras in different relative orientations, without the user having to directly measure or solve the parameters that define each configuration.

ÍNDICE

1. INTRODUCCIÓN

1.1. CONTEXTO Y MOTIVACIÓN DEL PROYECTO

Décadas atrás, la aparición del ratón y otros dispositivos similares, que permitían un seguimiento perceptualmente analógico de ciertos movimientos humanos, coincidió con los inicios de la popularización de la informática, y contribuyó significativamente en éstos. El salto en intuición de manejo no tenía precedentes en un tiempo en el que dominaba la idea de que la naturaleza discreta de los ordenadores debía ser palpable. Desde entonces, ha estado en el aire la posibilidad de una posible conversión de este dispositivo a las tres dimensiones, y a día de hoy ha habido ya numerosas iniciativas inspiradas en ella. Por un lado, las más populares son las que han añadido una tercera variable de entrada a dispositivos ya existentes, pero de forma que la naturaleza de su control es distinta. Ejemplos de esto son la famosa rueda de scroll en los ratones, en sus variantes de una y dos dimensiones, joysticks digitales dentro de mandos u otros joystick de mano, los ratones 3D diseñados específicamente para manejo de programas CAD (*Computer Aided Design*), y un etcétera que consiste básicamente en la unión de dos dispositivos independientes en uno, de forma que ambos sean manejables al mismo tiempo por el usuario. Estas variantes, aunque inmensamente útiles para ciertas aplicaciones, no sirven para una interactuación tridimensional "única" del usuario con la máquina, en el sentido de que al codificar dimensiones por medios distintos, no pueden registrar completamente un movimiento humano en el espacio. Por otro lado, se han desarrollado diversos sistemas que sí incluyen esta capacidad mencionada, pero están aún limitados a usos muy concretos.

Si se hace un análisis sobre las posibles causas, se hace evidente que con todas las ventajas que una entrada de este tipo ofrezca al usuario, hay inconvenientes prácticos. Principalmente, que el manejo de dispositivos que requieren un movimiento en dos dimensiones es más cómodo para el usuario en sesiones prolongadas, porque permite reposar la parte del cuerpo implicada, generalmente brazos y manos. El movimiento tridimensional, sin embargo, produce más cansancio en el usuario al cabo de un rato. Además, la mayor facilidad que tiene la mente humana para resolver problemas y movimientos en el plano frente a los del espacio también puede tener influencia, ya que si la gestión de un conjunto de objetos virtuales se puede hacer eficientemente en dos dimensiones, no tiene sentido complicarlo a una interfaz tridimensional sólo en aras de la vistosidad. Por todo ésto, aunque algunos sistemas desarrollados hasta la fecha como auténticos "ratones tridimensionales" hayan tenido prestaciones de precisión, retardo, portabilidad y coste aceptables, es normal que no hayan sustituido, ni vayan a sustituir a corto plazo, a la comodidad de los dispositivos señaladores de dos dimensiones. Sin embargo, sí pueden recibir una demanda creciente para las siguientes aplicaciones:

- Manejo de interfaces de usuario complejas en Sistemas Operativos, gestión de ficheros, acceso a bases de datos, etc.

- Introducción de datos en software especializado como simuladores, programas CAD, pizarras tridimensionales, etc.

- Aplicaciones de realidad virtual.

- Alternativa de los métodos de entrada existentes para personas con una discapacidad que impida, por ejemplo, un movimiento preciso de la mano y los dedos pero no del brazo.

Por otro lado, el reciente abaratamiento de los dispositivos de captación de imágenes, y la aparición de hardware con suficiente potencia de cálculo como para realizar tareas de procesado de imágenes en tiempo real incluso en dispositivos portables de bajo consumo, está haciendo viables nuevas alternativas para interfaces de entrada en ordenadores basadas en procesado de imágenes.

Por ejemplo, dentro de la evolución de los mismos ratones, uno de los saltos más palpables se produjo con la popularización de la tecnología óptica, en la que un hardware integrado en el dispositivo era capaz de seguir el movimiento del mismo haciendo comparaciones de fotografías monocromáticas sucesivas, tomadas por una pequeña cámara orientada hacia la superficie sobre la que se deslizara. Esto consiguió evitar los problemas mecánicos derivados de la tecnología anterior, donde la rotación de una bola se leía cuantificando por separado el giro de dos ejes perpendiculares pegados a ella.

Pero aunque la naturaleza de las imágenes es esencialmente bidimensional, también pueden emplearse para extraer información espacial. Una técnica, por ejemplo, permite localizar en cada instante un objeto en el espacio disponiendo de una sola imagen, empleando el tamaño de este en ella para obtener la información de profundidad. En esta técnica, sin embargo, para objetos pequeños y cámaras de prestaciones normales, la precisión se degrada rápidamente cuando el objeto se aleja de la cámara, ya que el tamaño del objeto en la imagen es inversamente proporcional a la distancia del objeto a la cámara, y por tanto el incremento de la primera magnitud en función de la segunda se hace pequeño rápidamente, volviendo el resultado susceptible al ruido, las distorsiones, y a las limitaciones de resolución de la cámara.

Una técnica más avanzada es la visión estereoscópica, que aprovecha la diferencia de perspectiva entre las imágenes tomadas por dos cámaras al mismo objeto en el mismo instante para inferir su posición en el espacio con respecto a dichas cámaras. La visión estereoscópica es una implementación particular del método geométrico más general conocido como triangulación, donde en cada instante se emplea información relativa a ángulos entre el punto a localizar y otros de referencia para situar al primero con respecto a los últimos. En el caso de la visión estereoscópica, los puntos de referencia desde los que se toman los ángulos son los centros de las cámaras, y los ángulos se toman por el punto de incidencia del rayo de luz que pasa por dicho centro e interseca una superficie fotosensible.

Conviene distinguir la triangulación de otra técnica llamada trilateración, donde se emplean las distancias a puntos de referencia en lugar de ángulos, y donde para el caso particular del espacio tridimensional, el número mínimo de referencias es tres en lugar de dos. En el caso de la triangulación, el tratamiento matemático pasa por resolver la intersección de variedades lineales en el espacio; rectas y planos si se localiza en el

espacio, mientras que en la trilateración son variedades cuadráticas, que serán esferas o hiperboloides dependiendo de que se empleen distancias absolutas o relativas (en este último caso, el método es llamado multilateración, y el número mínimo de referencias es una más). [1] Actualmente hay muchas implementaciones de este método, de entre los cuales el caso más conocido sea seguramente los sistemas de navegación por satélite, donde se emplea el retardo de señales de radio para extraer la información sobre distancias. Sin embargo, la discusión sobre qué naturaleza de señal, qué componentes físicos, frecuencias y técnicas de procesamiento de señal serían necesarias para obtener el rango de distancias y la precisión requeridas para esta aplicación queda fuera del interés de este texto.

[1] Es posible también localizar un punto en función de otros tres combinando información de distancias (mutuas entre los tres) y ángulos. Un método de este tipo se explicará en el apartado (3.2.1) "Fase 2 de calibrado", y se aplicará en la implementación final.

1.2. OBJETIVOS PRINCIPALES

El objetivo de este proyecto es discutir, diseñar e implementar un **sistema de seguimiento en tiempo real de la posición en el espacio de un objeto específico, para la entrada de datos en un ordenador, empleando visión estereoscópica (Objetivo 1)**. Tiene que ofrecer un rango de distancias y una precisión que permita una fidelidad aceptable en el seguimiento del objeto localizable cuando éste esté sujeto a los movimientos de la mano del usuario. De esta forma, podrá usarse como **apuntador o más en general, indicador de la posición de otro objeto como la extremidad de un usuario (Objetivo 2)**.

En general, la localización del objeto se realizará a través de un punto específico en él, referido en adelante como punto localizable, que será fácilmente detectable en las imágenes. Un objetivo adicional **(Objetivo A1)** es que el sistema pueda trabajar con diversos puntos localizables simultáneamente, lo cual puede servir para detectar la orientación del objeto además de su posición (si este tiene dos o más puntos localizables), o para localizar más objetos. Otro objetivo adicional **(Objetivo A2)** es que el sistema pueda funcionar con más de dos cámaras, lo cual añadiría redundancia y evitaría errores cuando un punto quedara ocluido con respecto a una cámara, o fuera de su rango de visión. Estos objetivos adicionales se abordarán cuando el sistema cumpla los anteriores, y se tratarán en un apartado independiente de la memoria.

Otro objetivo principal es que el sistema consistirá principalmente en una pieza de software diseñada para funcionar sobre **hardware de propósito general (Objetivo 3)**. Los motivos son la alta disponibilidad y bajo coste de este, que permitirán que cualquier usuario pueda emplear el sistema instalando el software en una computadora personal, y adquiriendo el mínimo hardware posible. De la mano de este objetivo está el que el sistema sea **suficientemente autónomo como para no requerir conocimientos especializados por parte del usuario, y ofreciendo un manejo simple (Objetivo 4)**.

1.3. DESCRIPCIÓN DEL DOCUMENTO

En el capítulo 2 “Estudio previo” se comenzará describiendo los principios teóricos generales de la visión estereoscópica y la terminología que se empleará a lo largo del documento. Posteriormente se describirá un método teórico extraído de la literatura existente, proponiéndose como candidato para alcanzar los objetivos buscados. Se mencionarán de la misma forma ejemplos de implementaciones existentes, tanto sistemas completos como librerías, y se discutirá su posible utilización, descartándose finalmente en favor de una implementación de elaboración propia.

A continuación, en el capítulo 3, “Desarrollo del proyecto”, se establecerán varios apartados. El primero valorará los objetivos planteados, sus consecuencias tanto por separado como en combinación, y definirá las características del sistema que sean necesarias para alcanzar un equilibrio en la consecución de estos. El segundo, con las características generales ya definidas, dividirá el sistema en módulos, y entrará en una descripción pormenorizada de los algoritmos y métodos matemáticos empleados para que cada uno cumpla sus requisitos de entrada y salida en las condiciones de eficiencia esperadas. El tercero especificará detalles de implementación referidos a lenguaje y arquitectura empleados, y sus repercusiones más importantes.

Por último, en el capítulo 4, “Conclusiones y posibles mejoras”, se valorarán los resultados finales en base a las pruebas realizadas sobre el sistema implementado, y se hará un análisis crítico del proyecto, exponiendo las posibles mejoras. También se considerarán aplicaciones alternativas de partes concretas del trabajo realizado.

Aunque la mayoría de operaciones de procesado de imágenes puede reducirse a matemáticas generales (teoría de conjuntos, transformaciones en el plano, operadores sobre funciones discretas), la terminología empleada en el campo, por motivos de brevedad, suele ser bastante específica. En este documento se hace uso de ella en los subapartados (3.2.3) “Fase 1 de calibrado” y (3.2.5) “Fase 1 de localización en tiempo real”, pero no se introduce explícitamente como se hace con la de la geometría epipolar. Si el lector no está familiarizado con esta terminología, puede hacer uso del breve glosario que se ha incluido al inicio del primero de los dos apartados, donde se ofrecen referencias externas detalladas.

Aunque hay algunos subapartados con una temática relativamente independiente con respecto a otros, se recomienda una primera lectura ordenada, ya que algunas particularidades de notación y nomenclatura se han introducido progresivamente, además de que se hacen alusiones frecuentes a cuestiones comentadas en subapartados anteriores. Sin embargo, si se desea una lectura más rápida, sin interés por la justificación detallada de ciertas decisiones, se puede obviar la lectura de dos puntos extensos: el subapartado (2.1.5) “Calibrado mediante coordenadas homogéneas”, y la sección “Análisis de métodos iterativos para la resolución de sistemas de ecuaciones no lineales” del subapartado (3.2.1) “Fase 2 de calibrado”.

2. ESTUDIO PREVIO

2.1. MODELO GENERAL

Modelo pinhole de una cámara

El modelo más simple para una cámara es aquel en el que, de todos los rayos de luz de una escena, se seleccionan aquellos que pasan por un punto, y de la intersección de éstos con una superficie plana se obtienen los puntos de la imagen. En el modelo más simple, el punto consiste en un pequeño agujero, de donde proviene el término *pinhole*. Si la imagen está en escala de grises o es multibanda, es cuestión de añadir una variable de intensidad a cada rayo, que se traduce en una en cada punto de intersección, y de repetir el proceso para cada banda. Este modelo general es independiente del fenómeno físico empleado para registrar la intersección del rayo con la superficie, pudiendo ser ésta desde una película fotosensible analógica a un CCD (*Charge-Coupled Device*) o un fotocátodo.

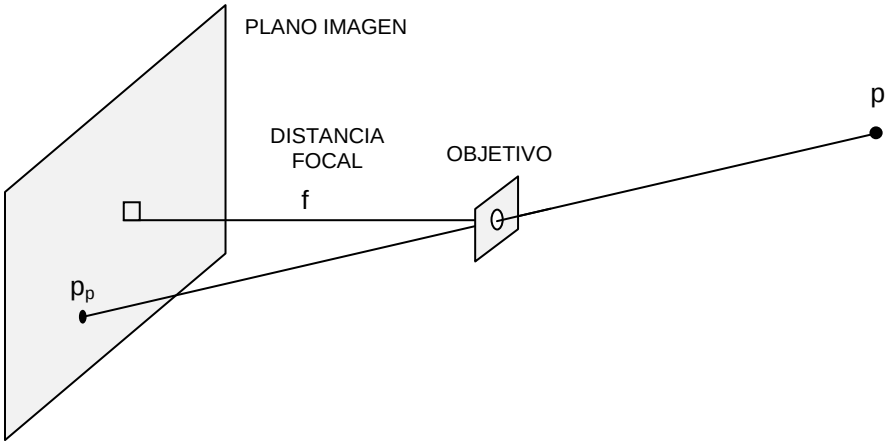


Figura 1.1

Si bien en los dispositivos físicos la superficie plana o sensor está detrás del punto objetivo, matemáticamente se puede modelar con la superficie tanto delante como detrás del objetivo, obteniendo los mismos resultados sin más que invertir las coordenadas de la imagen. Por otro lado, al tratar con la información de las cámaras digitales, hay que tener en cuenta que la mayoría de estas, aunque cumplen la disposición física comentada, ofrecen el resultado invertido equivalente al caso del plano delante, ya que perceptualmente resulta más natural al ser humano. Por sencillez, a partir de ahora se empleará un modelo matemático de la cámara con el plano delante del objetivo.

Una generalización de lo anterior nos hace darnos cuenta de que los resultados siguen siendo los mismos si se sitúa el plano ya no sólo a distancias opuestas en signo del objetivo, sino a una distancia arbitraria, y se reescalan proporcionalmente las

coordenadas de la imagen. Esta ambigüedad no es más que una versión en el espacio tridimensional del principio de semejanza entre triángulos, pero nos lleva a la necesidad de emplear una forma precisa de expresar la distancia entre punto y plano, conocida en óptica y fotografía como Distancia Focal (que tiene relación con la distancia focal de una lente, ya que como se verá poco más adelante, la labor del punto objetivo suele estar desempeñada por una lente real).

Dado que aunque el modelo matemático se refiere a un plano de extensión infinita, el sensor es finito, de forma rectangular, y cada punto de la imagen puede expresarse en relación a las dimensiones de dicho sensor, la ambigüedad puede resolverse especificando tanto la distancia entre el objetivo y el sensor como las dimensiones de este último. En el campo de la fotografía, por cuestiones históricas, ha solido expresarse la distancia focal asociada a un sensor de 35mm (cuyas dimensiones son 36mm x 24mm), aunque este no fuera siempre de dicho tamaño. Así, para conocer el campo de visión de una cámara en una configuración determinada, que formalmente se expresaría como un ángulo sólido en estereorradianes, en lugar de especificar el par (dimensiones del sensor, distancia focal), es suficiente expresar la distancia focal que daría el mismo campo de visión si el sensor fuese de 36mm x 24mm. Esta distancia focal es conocida como EFL (*35 mm Equivalent Focal Length*).

En este documento y en el código fuente de la implementación asociada, por simplicidad en los cálculos y salvo que se exprese lo contrario, se trabajará con la distancia focal asociada a un sensor cuyas dimensiones sean, en milímetros, equivalentes a la resolución de la imagen que está dando (no necesariamente la resolución nativa). Por ejemplo, si se están tomando imágenes QVGA (320x240 píxeles) de una cámara, se trabajará con la distancia focal asociada a un sensor de 230mm x 240mm. Estas dimensiones no son realistas para un CCD comercial, pero por lo que se ha visto antes, los resultados son matemáticamente equivalentes. Por corrección, a partir de ahora se relegará el término “sensor” para el dispositivo físico real, y se hablará de “plano imagen” para la región del plano matemático cuyas dimensiones se consideren asociadas a la distancia focal. Además, por homogeneidad, todas las medidas de distancia, salvo que se exprese lo contrario, estarán expresadas en milímetros.

Si se considera un sistema de referencia ajustado a la cámara, como en la figura, la relación matemática entre los puntos del espacio y de la imagen puede expresarse de la siguiente forma:

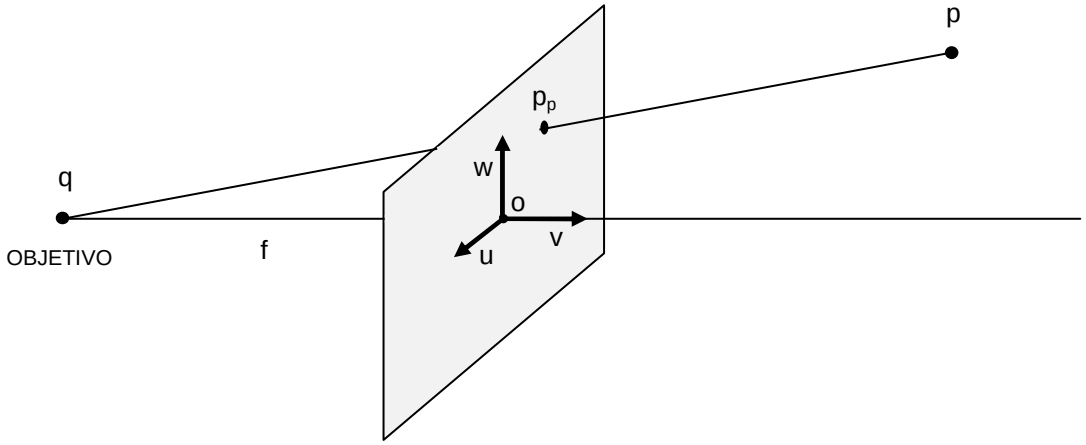


Figura 1.2

$$\{p_p\}_1 = \frac{f \cdot \{p\}_1}{f + \{p\}_2}$$
$$\{p_p\}_2 = 0 \qquad (1.1)$$
$$\{p_p\}_3 = \frac{f \cdot \{p\}_3}{f + \{p\}_2}$$

De ahora en adelante, se emplearán la notación *(u,v,w)* para la base de referencia, y $\{p\}_i$ para acceder a la componente *i* del punto o vector *p*. El punto objetivo se denotará con *q*. La notación de corchetes no es ortodoxa y puede parecer innecesariamente farragosa, pero será útil según el escenario se complique y el número de subíndices empiece a crecer.

El origen de las coordenadas *o* puede tomarse en el centro del plano imagen o en el objetivo, y la elección de la dimensión a lo largo de la cual se extiende el eje óptico es arbitraria, provocando sólo cambios fácilmente revertibles en las ecuaciones. Para esta sección del documento se ha empleado esta convención por coherencia con Duda y Hart (1973), ya que otros apartados del estudio previo describirán más adelante métodos extraídos de esta misma fuente. Sin embargo, en el apartado de "desarrollo del proyecto" se elegirá otra convención más adecuada al método finalmente empleado finalmente en la implementación.

Nótese que con lo expresado sobre el tamaño del plano imagen, y el sistema de referencia ajustado a la geometría de la cámara, la única información necesaria para simular una cámara (es decir, para poder calcular proyecciones de puntos tridimensionales) es la distancia focal. Si el sistema de referencia con el que se expresa la situación de *p* no está ajustado de ninguna forma a la geometría de la cámara, ha de disponerse de más información relativa a la posición y orientación de la cámara. En términos de información, sería suficiente con disponer de las ecuaciones del plano imagen y las coordenadas del punto objetivo para calcular la recta formada por el punto a proyectar y el objetivo, y posteriormente la intersección de esta con el plano imagen.

Sin embargo, los puntos resultantes, todos contenidos en un plano imagen que no es un caso trivial como el XZ de antes, están expresados con tres coordenadas, y su conversión a las dos coordenadas típicas en las que se expresa una imagen requiere de un movimiento rígido en el espacio, compuesto por una traslación y una rotación (si ambos sistemas de referencia son ortonormales, lo cual se da por supuesto). Este movimiento viene a ser el que caracteriza la posición (por la traslación) y la orientación (por la rotación) de la cámara en el espacio, y es suficiente por sí solo para caracterizar al propio plano imagen y al propio punto objetivo en función de los correspondientes a la cámara alineada con el sistema de referencia, teniendo f. Por tanto, con conocer este movimiento rígido, expresable matricialmente, y la distancia focal, es suficiente para caracterizar una cámara mediante el modelo pinhole en cualquier posición u orientación para una referencia del espacio dada.

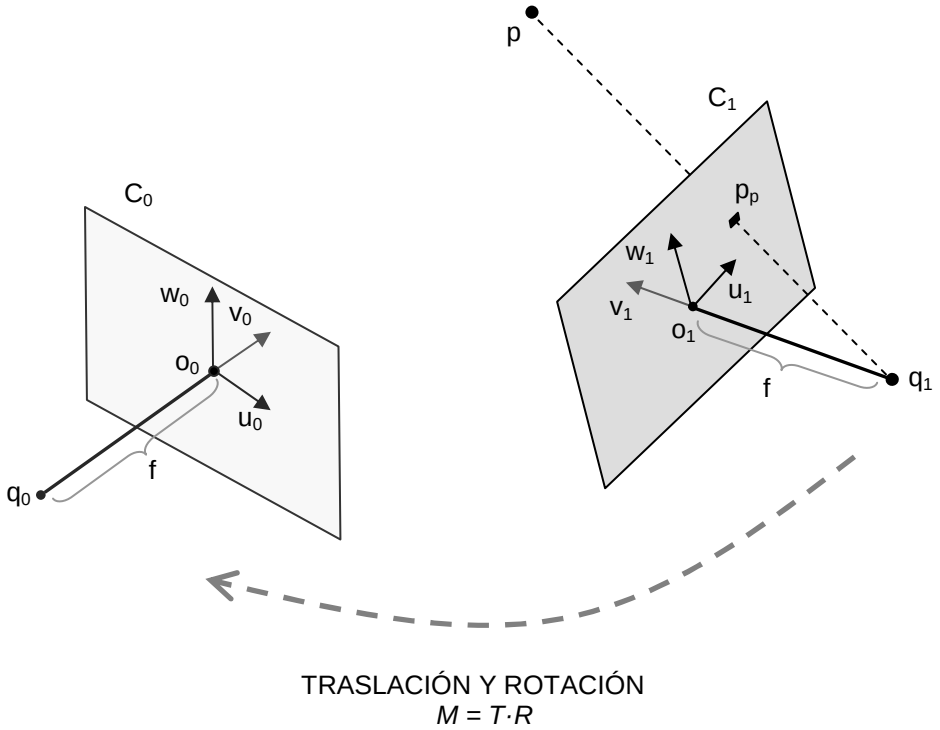


Figura 1.3

Es conveniente hacer hincapié en que aunque se habla de sistemas de referencia arbitrarios, se está haciendo una restricción a los rectangulares, es decir aquellos cuya base asociada es ortonormal (ortogonal y de vectores unitarios). Salvo por imposiciones externas, es preferible trabajar con este tipo de referencia, por diversos motivos: las transformaciones pueden interpretarse como movimientos rígidos compuestos de una traslación y una rotación en el espacio y es muy sencillo construir uno adaptado la geometría de cualquier cámara, como puede verse en la figura 1.3.

A continuación se detalla cada elemento de la figura 1.3, aprovechando para introducir cierta terminología, en parte extraída de la Duda y Hart (1973), y en parte creada ad hoc para una mayor claridad del texto en apartados siguientes.

o_0, u_0, v_0, w_0 Sistema de referencia empleado para representar los elementos de la escena. Dada una f, estará asociado a una cámara que puede llamarse "canónica" (C_0), que coincidirá con el modelo de la cámara empleada C_1 si la escena se describe en función del sistema de referencia asociado a ella, que es el caso sencillo introducido antes.

o_0 *(0,0,0)* Origen del sistema de referencia global empleado.

u_0 *(1,0,0)* Vector de la base del sistema de referencia global empleado.

v_0 *(0,1,0)* "

w_0 *(0,0,1)* "

q_0 *(0,-f,0)* Objetivo de C_o, también llamado "punto objetivo" o "pinhole lens".

o_1, u_1, v_1, w_1 Sistema de referencia ortonormal asociado a la cámara en posición arbitraria C_1

o_1 Objetivo de C_1

p Posición del punto a proyectar en el espacio, que más adelante será a localizar, y se hará referencia a él como "punto localizable", "punto tridimensional" o "primitiva 3d". Como ya se mencionó, no debe confundirse con el objeto localizable, que puede contener uno o más puntos localizables.

p_p Posición de la proyección de p en la imagen de la cámara C_1. Se hará referencia a él como "punto imagen" o "primitiva 2d". Sus coordenadas en el sistema de referencia asociado a C_1 son del tipo *(x,0,z),* siendo x y z las coordenadas x e y de la imagen.

M_1 Transformación rígida que convierte de un sistema de referencia a otro. En concreto:

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \{o_1\}_1 & \{u_1\}_1 & \{v_1\}_1 & \{w_1\}_1 \\ \{o_1\}_1 & \{u_1\}_2 & \{v_1\}_2 & \{w_1\}_2 \\ \{o_1\}_1 & \{u_1\}_3 & \{v_1\}_3 & \{w_1\}_3 \end{bmatrix} \in \mathrm{M}_{4\times 4}(\Re)$$

realiza la transformación del sistema de referencia de la cámara C_1 al de la cámara canónica.

Distorsiones sobre el modelo pinhole

Los sistemas reales que cumplen literalmente el modelo pinhole, que en castellano reciben el nombre de cámaras esteneopeicas y tienen por objetivo un simple agujero en una superficie opaca, tienen muchas limitaciones prácticas. La cantidad de luz que puede pasar por un agujero lo suficientemente pequeño como para dar una resolución aceptable en la imagen es muy pequeña, por lo que para materiales de fotosensibilidad media se requieren tiempos de exposición muy largos para cada fotograma, con los problemas de resolución temporal que ello conlleva. Además, agujeros demasiado pequeños pueden revelar problemas de difracción, por acercarse el diámetro a la

longitud de onda a la que es sensible la superficie, y de oscurecimiento en los laterales si el diámetro se acerca a la anchura del material opaco sobre el que se forma el agujero. Por esta razón, la gran mayoría de las cámaras usan una lente como objetivo, que dirige haces de luz de diámetro mayor a puntos sobre la superficie fotosensible, provocando el mismo efecto de proyección que el agujero puntual, pero permitiendo una mayor intensidad en cada punto proyectado. Sin embargo, las lentes provocan una serie de distorsiones sobre los resultados ideales del modelo pinhole, de entre las cuales cabe destacar la distorsión radial, por ser la que tiene mayores consecuencias sobre la geometría de la imagen y la que suele tenerse en cuenta para aplicaciones de visión artificial donde se requiere una localización muy precisa. Si se considera que en la imagen obtenida $x = \{p_p\}_1$, $y = \{p_p\}_2$, su expresión matemática es la siguiente:

$$\begin{aligned} x_{real} &= (1 + k_{1x}d^2 + k_{2x}d^4 + k_{3x}d^6 + ...) \cdot x_{teórica} \\ y_{real} &= (1 + k_{1y}d^2 + k_{2y}d^4 + k_{3y}d^6 + ...) \cdot y_{teórica} \end{aligned} \qquad d = \sqrt{x_{teórica}^2 + y_{teórica}^2} \tag{2.1}$$

La distorsión radial, determinada por los coeficientes *k*, se debe a la diferencia entre una lente parabólica ideal y una esférica, de menor coste de fabricación. Para cámaras con una alta distorsión, bien por ser baratas o por tener distancias focales muy cortas (ojo de pez) se suelen considerar hasta coeficientes de orden 3. Sin embargo, en la mayoría de los casos, es aceptable la simplificación que considera sólo los coeficientes asociados al cuadrado de la distancia, y éstos iguales para ambos ejes, de modo que la expresión queda en:

$$\begin{aligned} x_{real} &= (1 + k \cdot (x_{teórica}^2 + y_{teórica}^2)) \cdot x_{teórica} \\ y_{real} &= (1 + k \cdot (x_{teórica}^2 + y_{teórica}^2)) \cdot y_{teórica} \end{aligned} \tag{2.2}$$

En general, el coeficiente de distorsión es mayor que cero para distancias focales largas, generando una distorsión de tipo cojín sobre una figura originalmente cuadrada, y menor que cero para las cortas, provocando un efecto barril. También, el módulo de la distorsión tiende a ser mayor para distancias focales cortas, como se puede apreciar en las imágenes tomadas con un objetivo ojo de pez.

Otro tipo de distorsión notable, que por ello se incluye en algunas modelizaciones, es la tangencial. A diferencia de la anterior, ésta no es intrínseca a la lente, sino que se debe a que, en la práctica, el sensor no se sitúa en el plano perpendicular al eje óptico de la lente, si no que existe cierta inclinación inherente a la imperfección del proceso de fabricación. Esta distorsión se caracteriza con dos coeficientes, correspondientes al vector gradiente del plano que contiene al sensor real, con respecto al plano perpendicular ideal.

En algunas modelizaciones se considera también el hecho de que en la práctica el centro óptico de la imagen (el punto a partir del cual se calcula la anterior distorsión) no coincide con el centro de la imagen en sí, por una imperfección en la situación relativa de la lente y el sensor. Esto se traduce en dos nuevas variables c_x, c_y , que son las coordenadas sobre la imagen del centro de las distorsiones.

Proyecciones inversas y geometría epipolar

Todo lo mencionado en el apartado del Modelo Pinhole hace referencia a modelar matemáticamente la naturaleza proyectiva de una cámara, de forma que si se dispone de p , puede simularse la proyección que realizaría una cámara real y obtener p_p. Pero el mismo modelo puede emplearse para realizar el proceso inverso, y disponiendo de una imagen tomada por una cámara real en la que pueda localizarse p_p , y con información acerca de los parámetros de dicha cámara (f, M), puede obtenerse información acerca de la situación de p en el espacio, que viene a ser el objetivo desde el principio. Por todo lo dicho anteriormente, el modelo sólo debería consistir en la distancia focal y opcionalmente los coeficientes de las distorsiones consideradas, si se admite que las posiciones tridimensionales van a estar expresadas en el sistema de referencia asociado a esa misma cámara.

Sin embargo, es inmediato ver que la información sobre la proyección de un solo punto en una cámara no es suficiente para localizar dicho punto. El resultado es una recta de posibles localizaciones del punto, que pasa por el objetivo y por el punto en el plano imagen. Si se define p con el sistema de referencia asociado a la cámara (C_1=C_0), y se ignoran las distorsiones, las ecuaciones de la recta son:

$$\{p\}_1 = \frac{\{p_p\}_1}{f}(\{p\}_2 + f) = \frac{\{p_p\}_1}{\{p_p\}_3}\{p\}_3 \tag{3.1}$$

Para localizar un objeto se necesita por tanto más información. Se puede resolver el grado de indeterminación a través del área generada por la proyección del objeto localizable (que incluye al punto localizable, pero tiene volumen y su proyección área). Para que esta técnica ofrezca resultados aceptables, el área de la proyección del objeto debe ser invariante a la rotación, es decir, una esfera, o si varía, que sea posible inferir la orientación a partir de la geometría de la proyección. Aún así, como ya se comentó en la introducción, la precisión de esta técnica se degrada rápidamente en cuanto el objeto es de tamaño mediano y se aleja algo de la cámara, ya que la razón

$$\frac{\Delta(area\_proyeccion)}{\Delta(dist)}$$

se aproxima a cero. Otra opción es emplear la proyección de varios puntos, junto con cierta información de estos en el espacio, como la distancia relativa entre ellos. Esta técnica, como también se comentó en la introducción, se tratará más adelante.

Cuando se conoce la situación relativa de dos cámaras, sus distancias focales y se supone un modelo pinhole para ellas, se puede emplear la geometría epipolar para determinar la posición de regiones puntuales en el espacio. En principio, es suficiente con buscar p_{p1} y p_{p2} exhaustivamente en las imágenes ofrecidas por C_1 y C_2 respectivamente, y hallar el corte entre las rectas generadas por cada cámara, pero la geometría epipolar proporciona una metodología que puede ahorrar cómputo y a la vez reducir errores en la tarea previa de la localización simultánea de distintas proyecciones del mismo punto tridimensional, conocida como “problema de correspondencia”.

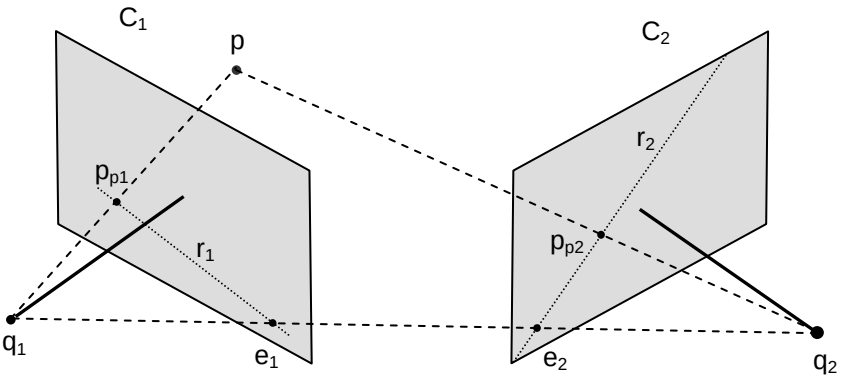


Figura 3.1

Existe una terminología específica para los elementos matemáticos involucrados en la geometría epipolar, pero la idea esencial se puede observar en la figura 3.1. Se dispone de dos cámaras cuyos parámetros (f_i, M_i / i=1,2) se conocen, usando un sistema de referencia ortonormal arbitrario (nótese que por ello no se emplea M_0). De momento se obviarán las distorsiones. Gracias a M_i, podemos calcular los puntos objetivo:

$$q_i = o_i - f \cdot v_i \tag{3.2}$$

Si se explora la imagen de C_1 hasta encontrar p_{p1} , basta calcular la recta intersección entre el plano dado por (q_1,q_2,p_{p1}), y el plano imagen de C_2 para buscar sobre ella el punto p_{p2} necesario para localizar p. A p_{p2} se le denomina conjugado de p_{p1}, el plano dado por (q_1,q_2,p_{p1}) se denota como "plano epipolar", y la recta resultante como "línea epipolar". El planteamiento es totalmente reversible, de modo que existen dos puntos conjugados y dos líneas epipolares conjugadas. La cuestión es que el uso de la línea epipolar de C_2 ahorra la re-exploración de una imagen entera por segunda vez, algo muy interesante teniendo en cuenta el alto coste computacional de la tarea, especialmente si la localización de p_{p2} no es obvia en la imagen (esto depende de multitud de condiciones relacionadas con el entorno físico en el que se toman las imágenes y las propiedades del objeto que contiene el punto que ocupa la posición de p, y se hablará de ello más adelante en el documento). En la práctica, la línea epipolar debe transformarse al sistema de referencia asociado a C_2 para que quede contenida en el plano imagen, creando un segmento acotado por sus límites, que será el que se emplee finalmente para determinar qué píxeles de la imagen de C_2 deben explorarse. Esto puede hacerse empleando $(M_2)^{-1}$, de la misma forma que debe haberse empleado M_1 para expresar p_{p1} en función del sistema de referencia arbitrario empleado, y así poder construir el plano epipolar junto con q_1 y q_2. En la práctica, se reduce el número de transformaciones necesarias si como sistema de referencia general se emplea el asociado a una cámara, que desde ese momento se convertirá en portadora de la referencia absoluta de toda la escena, y que por convención se denotará en este texto, cuando se use, como C_1 (de modo que $C_1 = C_0$). Además, se puede emplear el hecho de que cada uno de los puntos e_i

(llamados epipolos) está siempre está incluido en la línea epipolar de su correspondiente cámara C_i, y sus coordenadas en la imagen son invariantes mientras que se mantenga la configuración de la escena. Esto se puede aprovechar para reducir los requisitos de transformación de coordenadas de la línea epipolar a la transformación de un solo punto que pertenezca a ella y no sea e_i.

Por otro lado, el método se puede generalizar a más de dos cámaras (C_1, C_2, C_3,…), bien para añadir redundancia ante oclusiones de *p*, o para minimizar errores introducidos por distorsiones no modeladas (por MSE, por ejemplo). Esto ya se introdujo como objetivo opcional anteriormente.

Existe una técnica que se emplea cuando los ejes ópticos de ambas cámaras (considerando dos) son paralelos, las distancias focales iguales, y los puntos objetivo forman una recta perpendicular a ambos, llamada "línea base". Es decir, están en posiciones distintas, pero orientadas en la misma dirección y a la misma altura, y tienen una óptica equivalente. Si además se considera que la línea base es paralela al eje de coordenadas horizontales de los sensores de las dos cámaras (es decir, no están inclinadas lateralmente), resulta que las líneas de búsqueda de ambas imágenes son horizontales y están a la misma altura (es decir, considerando una resolución igual para ambas cámaras, las coordenadas de sus píxeles son equivalentes). Esto simplifica mucho el problema. Además, esta situación concreta permite un tratamiento muy simple de la posterior tarea de resolución de *p* por ecuaciones lineales, basándose en mera semejanza de triángulos. En concreto, sólo es necesario calcular la diferencia entre las coordenadas horizontales de las dos proyecciones del mismo punto, a la que se denomina "disparidad", para hallar la coordenada de profundidad de *p* mediante una expresión muy simple. Resulta que ambas magnitudes son inversamente proporcionales, con el factor de proporcionalidad dado por magnitudes constantes en la escena (distancia focal, distancia entre cámaras). Las otras dos coordenadas de *p* se pueden hallar con expresiones similares. Sin embargo, considerando los objetivos 3 y 4, no puede asumirse la disponibilidad de la instrumentación necesaria para calibrar físicamente las cámaras, de modo que esta técnica no es útil para este caso. Por otro lado, el enfoque consistente en realizar una rectificación de los datos obtenidos por una cámara para convertirlos a los que obtendría una situada paralelamente a la otra, de forma que pueda emplearse el método descrito antes, no parece ofrecer ventajas computacionales sobre el aquí expuesto ya que el proceso de reproyección que implica la rectificación añade una complejidad que diluye la mencionada simplicidad.

Sin embargo, a pesar de todo lo dicho para el caso general, el empleo de la línea epipolar puede ser problemático, porque si el modelo de las cámaras no se ajusta bien a la realidad, sea porque la posición, orientación, o distancia focal consideradas para alguna o todas las cámaras sufren de cierto error, o porque las distorsiones de la lente son notables (suelen serlo más en objetivos con distancia focal pequeña; i.e. gran angular), el punto a buscar puede quedar fuera de su recorrido. Emplear una anchura de búsqueda muy grande hace que se pierda la ventaja de menor coste computacional, y si se está trabajando con varios puntos simultáneamente, puede llevar a relacionar rectas correspondientes a puntos distintos, con los errores desproporcionados que ello conlleva. El coste computacional puede llegar a ser incluso mayor que en el caso de explorar cada imagen con un solo barrido independiente, en situaciones donde estén involucrados muchos puntos simultáneamente y anchos de búsqueda demasiado

generosos. Si algunas distorsiones ópticas han sido proporcionadas por el fabricante o inferidas por el usuario, se pueden aplicar sus efectos sobre la línea de búsqueda, deformándola, y posteriormente aplicar la transformación inversa sobre las primitivas halladas para compensar las distorsiones (mucho más liviano computacionalmente que compensarlas aplicando la transformación inversa directamente al raster). Pero la eficacia de este método depende de que las distorsiones existentes se deban mayoritariamente a las distorsiones modeladas, lo cual no tiene por qué suceder.

En adelante, y siguiendo la nomenclatura empleada habitualmente en la literatura, los parámetros que caracterizan el modelo de una cámara en una escena particular, que son necesarios para emplearla en visión estereoscópica, se dividirán en dos tipos: extrínsecos e intrínsecos. Los primeros son la posición y la orientación de la cámara con respecto a un sistema de referencia dado, que puede estar o no ajustado a otra cámara (si está ajustado a la misma cámara, estos parámetros son triviales). Los segundos consisten en la distancia focal y cualquier distorsión considerada que sea independiente de la posición u orientación particulares de la cámara.

Calibrado del sistema

Como se ha dicho en el apartado anterior, para emplear visión estereoscópica, han de conocerse los parámetros intrínsecos de las cámaras (distancia focal, coeficientes de distorsión opcionalmente) y extrínsecos (posición y orientación relativas entre cámaras).

Algunos fabricantes de cámaras incluyen información sobre la distancia focal y el tamaño del sensor real o equivalente, pero otros, especialmente para el caso de cámaras de bajo coste, no. En cuanto a los coeficientes de distorsión, cabe decir que en general sólo cámaras de alto coste comercializadas específicamente para aplicaciones de fotogrametría (de la cual la visión estereoscópica por ordenador se podría considerar una rama) incluyen información acerca de ellos. Aunque estas distorsiones no van a ser consideradas en este proyecto, lo anterior junto con el Objetivo 3 mencionado previamente en la memoria (emplear hardware de propósito general) nos lleva a que el sistema deberá disponer de una función para averiguar la distancia focal para los casos en los que esta no sea revelada por el fabricante.

En cuanto a los parámetros extrínsecos, de nuevo el Objetivo 3, junto con el Objetivo 4 (grado de autonomía elevado y posibilidad de uso sin conocimientos especializados), nos impiden el uso de maquinaria de precisión para calibrar la posición y orientación relativa de las cámaras, por lo que nuevamente el sistema deberá averiguar estos datos por sí mismo. En apartados posteriores del documento se entrará más en detalle en las consecuencias de cada objetivo marcado inicialmente.

Para ambas tareas, la estrategia más típica es fotografiar con cada cámara un patrón cuya configuración tridimensional es conocida (típicamente, consistente en puntos fácilmente distinguibles en las fotografías, y siendo conocidas las distancias relativas entre dichos puntos), y construir un sistema de ecuaciones cuya solución describa la proyección cónica que puede transformar esa configuración tridimensional en la configuración bidimensional detectada en la imagen.

Si se consideran las distorsiones radiales, el modelo es más complejo que el de una proyección cónica, y el método para resolver todos los parámetros más complicado. También es natural que se necesite un patrón de calibrado más complejo, ya que a mayor número de incógnitas, mayor número de ecuaciones son necesarias. Un método popular para este caso es el desarrollado en Tsai (1987), que resuelve el conjunto de todos los parámetros extrínsecos e intrínsecos (incluyendo el coeficiente de distorsión radial simplificada expuesto en la ecuación (2.2)). Para ello emplea un patrón compuesto por un número elevado de puntos, que podrán ser coplanares o no, aunque para cámaras con sensor CCD el método recomendado es el de los puntos coplanares. Este consta de dos etapas diferenciadas, en cada una de las cuales se resuelve un subconjunto disjunto de parámetros de una cámara: los tres ángulos que describen la rotación (*yaw, pitch, roll*)[2] y dos coordenadas de traslación en la primera etapa, y la tercera coordenada de traslación (la asociada al eje óptico en este caso), la distancia focal y el coeficiente de distorsión radial en la segunda etapa. Esta división en dos etapas se puede hacer debido a que, para un punto proyectado, es posible aislar cierta propiedad que es independiente de los valores de los parámetros que se resuelven en la segunda etapa, dependiendo sólo de los de la primera. Por tanto, esta propiedad de cada punto proyectado, con un número adecuado de puntos en el patrón, puede emplearse para la resolución de los parámetros de la primera etapa, obviando temporalmente los de la segunda. La ventaja de ésto es que evita la necesidad de aplicar un método de búsqueda no lineal sobre un espacio de dimensionalidad demasiado alta. La propiedad en concreto es la dirección del vector que une el centro de la imagen y el punto proyectado, que efectivamente se mantiene invariante para cualquier movimiento paralelo al eje óptico de la cámara, o cualquier cambio de la distancia focal o de la distorsión radial (si se supone que el centro óptico de la lente y el centro del sensor coinciden). A su vez, la segunda etapa, que se computa con los parámetros de la primera ya resueltos, se divide en dos subetapas: una que obvia la distorsión radial, y que arroja una solución aproximada cuya obtención es más sencilla, y otra que emplea dicha aproximación como punto de partida para una búsqueda iterativa hacia una solución más exacta.

Sin embargo, el sistema a construir en este caso, por lo expuesto en el Objetivo 2, no requiere de prestaciones elevadas de precisión en la localización en tiempo real del patrón (precisión que dependerá directamente del grado de realismo del modelo de cámara aplicado, y del método de calibrado empleado). Más adelante, en el apartado de "Características del sistema" se razonará detalladamente la conexión entre el Objetivo 2 y dicha permisividad en el error de localización del patrón, mientras se mantengan ciertas condiciones. Por el momento cabe adelantar que al no existir grandes requisitos de precisión, puede obviarse el efecto de la distorsión radial, simplificando considerablemente el problema de calibrado. Además, esto permitirá emplear métodos de calibrado que requieran patrones de calibrado más sencillos, como se verá en el apartado de "Descripción del sistema". Esta decisión, en cualquier caso, será revisada críticamente en el apartado de Conclusiones y mejoras de este proyecto.

[2] Existen distintas notaciones para designar los tres ángulos de Euler, que definen una rotación en el espacio de tres dimensiones. En este texto se adaptará la empleada en aeronáutica, de forma que *yaw* y *pitch* producen un movimiento horizontal y vertical respectivamente en la imagen de la cámara, y *roll* produce una rotación de la imagen. En otras palabras, un ángulo *roll* es el que define un giro de la cámara alrededor de su eje óptico.

Como consencuencia de esta decisión, a continuación se centrará la atención en un método de calibrado propuesto en *Pattern recognition and scene analysis* de Duda y Hart (1973), que no tiene en cuenta ningún tipo de distorsión óptica. Este método es mencionado también en Shuterland (1974), Yakimovski y Cunningham (1978), así como en la propia publicación de Tsai, y también es explicado de forma resumida en *Visión por computador* de Javier González Jiménez (2000). Se ha empleado como fuente la obra original de Duda y Hart, añadiendo y eliminando algunos puntos, y cambiando cierta notación. La conveniencia de usar este método u otro en el sistema a tratar, según los objetivos establecidos al principio, será valorada en el apartado de "Descripción del Sistema", en base a las conclusiones extraídas de su estudio detallado. Sin embargo, para permitir una lectura más rápida a quien prefiera centrar la atención en otros aspectos de esta memoria, se puede adelantar que el método será finalmente descartado en favor de uno de elaboración propia. Por tanto, el siguiente capítulo puede ser omitido de la lectura sin perjuicio para la comprensión de los demás.

Calibrado mediante coordenadas homogéneas

Para el caso del modelo pinhole se describirá a continuación el método propuesto por Duda y Hart (1973), que trata de describir la proyección de los puntos tridimensionales como una aplicación lineal, empleando coordenadas homogéneas. Conseguir esto puede servir para que la obtención de los parámetros intrínsecos y extrínsecos se pueda reducir también a la resolución de un sistema de ecuaciones lineales. Sin embargo, la técnica empleada tiene ciertos inconvenientes, como el de aumentar los grados de libertad y por consiguiente el error.

Aunque la transformación descrita en las ecuaciones (1.1), que es una proyección cónica, no es lineal (nótese que la coordenada segunda aparece en el denominador), puede interpretarse como lineal si se trabaja en coordenadas homogéneas, y así también puede expresarse en formato matricial. Hagamos que si $p = (\{p\}_1, \{p\}_2, \{p\}_3)^t$ es el punto del espacio tridimensional que va a ser proyectado, como ha venido siendo expresado hasta ahora, en coordenadas homogéneas se exprese como $\widetilde{p} = (h \cdot \{p\}_1, h \cdot \{p\}_2, h \cdot \{p\}_3, h)^t$, donde $h \in \Re$ es un valor cualquiera. A partir de ahora, se empleará la notación de la tilde para denotar el uso de coordenadas homogéneas. Con lo visto, para obtener el punto original han de dividirse las coordenadas homogéneas entre la nueva cuarta coordenada, y existen múltiples respresentaciones homogéneas del punto original.

Ahora, si consideramos la matriz:

$$F = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & \frac{1}{f} & 0 & 1 \end{bmatrix} \tag{5.1}$$

y si $p=[\{p\}_1,\{p\}_2,\{p\}_3]^t$ es un punto cualquiera en el espacio y $p_p=\begin{bmatrix}\frac{f\cdot\{p\}_1}{f+\{p\}_2} & 0 & \frac{f\cdot\{p\}_3}{f+\{p\}_2}\end{bmatrix}^t$ su proyección según las ecuaciones (1.1), resulta que:

$$F\cdot\widetilde{p}=F\cdot\begin{bmatrix}h\cdot\{p\}_1\\ h\cdot\{p\}_2\\ h\cdot\{p\}_3\\ h\end{bmatrix}=\begin{bmatrix}h\cdot\{p\}_1\\ h\cdot\{p\}_2\\ h\cdot\{p\}_3\\ \frac{h\cdot\{p\}_2}{f}+h\end{bmatrix}=\widetilde{p}_p \tag{5.2}$$

La última igualdad se debe a que:

$$\frac{h\cdot\{p\}_1}{\frac{h\cdot\{p\}_2}{f}+h}=\frac{f\cdot\{p\}_1}{f+\{p\}_2} \qquad \text{y} \qquad \frac{h\cdot\{p\}_3}{\frac{h\cdot\{p\}_2}{f}+h}=\frac{f\cdot\{p\}_3}{f+\{p\}_2} \tag{5.3}$$

Esto indica que el resultado de la aplicación lineal es la representación homogénea del resultado de la proyección en lo que a la primera y tercera coordenada respecta. La segunda no coincide, ya que según lo expuesto en las ecuaciones (1.1), que se basan en el modelo donde el plano imagen coincide con el plano XZ, esta debe ser igual a cero. Sin embargo, es algo que se puede obviar ya que lo importante en términos de información de los puntos de la imagen son la primera y tercera coordenadas, que juegan el papel de primera y segunda coordenadas de la imagen respectivamente. La razón de que no coincidan puede explicarse en términos de que al ser F no singular y tener la transformación inversa múltiples soluciones (la que asocia a cada punto de la imagen los posibles puntos tridimensionales que producirían tal punto imagen), debe haber un grado de libertad en la imagen de F para que se cumpla la definición de función que obliga a que dos imágenes distintas tengan orígenes distintos. En cualquier caso, no es importante en el tratamiento matricial de la proyección como herramienta para el calibrado, ya que no se manejará la proyección inversa.

Por tanto, se puede decir que las representaciones homogéneas del punto a proyectar y del punto proyectado cumplen una relación de transformación lineal. Sin embargo, es importante señalar otra cosa que sí es importante. Se puede entrever que las coordenadas homogéneas no eliminan realmente la necesidad de la división entre una de las coordenadas del resto de ellas, sino que la desplaza al momento de la transformación de coordenadas homogéneas a originales. Es más, no se puede afirmar que la relación de proporcionalidad (es decir, la cuarta coordenada) sea la misma antes y después de aplicar la transformación. Esto implica que si se quiere reescribir la ecuación (5.2) nombrando las componentes no homogéneas tanto en el punto tridimensional como en el punto imagen, deban usarse dos nuevos parámetros *h* distintos:

$$F \cdot \widetilde{p} = F \cdot \begin{bmatrix} h_0 \cdot \{p\}_1 \\ h_0 \cdot \{p\}_2 \\ h_0 \cdot \{p\}_3 \\ h_0 \end{bmatrix} = \begin{bmatrix} h_1 \cdot \{p_p\}_1 \\ h_1 \cdot \{p_p\}_2 \\ h_1 \cdot \{p_p\}_3 \\ h_1 \end{bmatrix} = \widetilde{p}_p \tag{5.4}$$

Tampoco se puede obtener h_1 en función de h_0 de forma independiente del punto tridimensional que se está proyectando. Las consecuencias de esto se verán más adelante.

Con lo expuesto hasta ahora, la matriz de la transformación lineal describe sólo los parámetros intrínsecos de la cámara pinhole, o sea la distancia focal f. Para que describa los parámetros extrínsecos, es decir una posición y una orientación arbitrarias, pueden aplicarse la traslación y rotación opuestas a los puntos del espacio antes de ser proyectados. Esto se consigue componiendo la matriz F con una matriz T de traslación y una R de rotación en el espacio:

$$T = \begin{bmatrix} 1 & 0 & 0 & -\{t_0\}_1 \\ 0 & 1 & 0 & -\{t_0\}_2 \\ 0 & 0 & 1 & -\{t_0\}_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\cos(\varphi)\sin(\theta) & \cos(\varphi)\cos(\theta) & \sin(\varphi) & 0 \\ \sin(\varphi)\sin(\theta) & -\sin(\varphi)\cos(\theta) & \cos(\varphi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.5}$$

Esto es similar a lo expuesto en el apartado donde se explica el modelo Pinhole, aunque en este caso la coordenada homogénea se coloca en cuarto lugar en vez de en el primero (esta es una diferencia trivial y no se ha considerado necesario cambiarlo). Sin embargo, debido al efecto de la matriz *F*, ahora esta coordenada no va a ser necesariamente igual a 1 en los vectores, por lo que las cosas serán distintas.

Por otro lado, en Duda y Hart no se emplea una matriz que incluya los tres grados de libertad posibles en una rotación en el espacio de tres dimensiones (*yaw, pitch, roll*), sino que se asume que la cámara siempre mantendrá su eje transversal en situación perfectamente horizontal, igualando así el ángulo *roll* a cero y eliminando su incógnita. El planteamiento se puede generalizar sin que afecte a todo el desarrollo que sigue, ya que como se verá más adelante, se van a tener que aumentar irremediablemente los grados de libertad del problema, pero se ha mantenido el planteamiento original de la fuente por simplicidad.

De esta forma, $C = FRT$ constituye la matriz de una transformación lineal que representa la proyección de una cámara (trabajando en coordenadas homogéneas, y olvidándonos de la segunda coordenada), y que es definida por los parámetros intrínsecos y extrínsecos de la cámara modelada como pinhole. Por obviar la segunda coordenada del punto imagen, puede hacerse lo mismo con la segunda fila de *C*, y por tanto reducirse la matriz a dimensiones 3 x 4. Por tanto, la "ecuación de captación de la cámara" puede denotarse en su estado matricial definitivo como:

$$\widetilde{p}_p = \begin{bmatrix} h_1 \cdot \{p_p\}_1 \\ h_1 \cdot \{p_p\}_3 \\ h_1 \end{bmatrix} = C \cdot \begin{bmatrix} h_0 \cdot (\{p\}_1)_{RX} \\ h_0 \cdot (\{p\}_2)_{RX} \\ h_0 \cdot (\{p\}_3)_{RX} \\ h_0 \end{bmatrix} = C \cdot (\widetilde{p})_{RX} \tag{5.6}$$

Si se desarrolla el producto, puede comprobarse que:

$$C = FRT = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 & -\{t_0\}_1 \cos(\theta) - \{t_0\}_2 \sin(\theta) \\ \sin(\varphi)\sin(\theta) & -\cos(\theta)\sin(\varphi) & \cos(\varphi) & \{t_0\}_2 \cos(\theta)\sin(\varphi) - \{t_0\}_3 \cos(\varphi) - \{t_0\}_1 \sin(\varphi)\sin(\theta) \\ -\dfrac{\cos(\varphi)\sin(\theta)}{f} & \dfrac{\cos(\varphi)\cos(\theta)}{f} & \dfrac{\sin(\varphi)}{f} & \dfrac{\{t_0\}_1 \cos(\varphi)\sin(\theta)}{f} - \dfrac{\{t_0\}_2 \cos(\varphi)\cos(\theta)}{f} - \dfrac{\{t_0\}_3 \sin(\varphi)}{f} + 1 \end{bmatrix}$$

La notación $(\cdot)_{RX}$ en (5.6) indica que las coordenadas del punto en el espacio pueden estar expresadas en cualquier sistema de referencia, ya que la transformación incluye la traslación y la rotación. Ello, claro, mientras dicha referencia emplee una base ortonormal, ya que la rotación sólo puede convertir entre sistemas de referencia de este tipo.

Llegados a este punto, está claro que el problema de calibrar las cámaras de un sistema completo es el de averiguar los coeficientes de la matriz C de cada cámara, empleando para ello la información dada por una colección suficiente de *n* puntos tridimensionales, formando un patrón conocido, y sus correspondientes puntos imagen en cada cámara, formando lo que ésta captaría de dicho patrón (recordemos, desde cualquier distancia y punto de vista, gracias a la introducción de T y R). En forma matricial:

$$\widetilde{P}_p = \begin{bmatrix} h_1 \cdot \{p_{p1}\}_1 & \cdots & h_n \cdot \{p_{pn}\}_1 \\ h_1 \cdot \{p_{p1}\}_3 & \cdots & h_n \cdot \{p_{pn}\}_3 \\ h_1 & \cdots & h_n \end{bmatrix} =$$

$$= C \cdot \begin{bmatrix} h_0 \cdot (\{p_1\}_1)_{RX} & \cdots & h_0 \cdot (\{p_n\}_1)_{RX} \\ h_0 \cdot (\{p_1\}_2)_{RX} & \cdots & h_0 \cdot (\{p_n\}_2)_{RX} \\ h_0 \cdot (\{p_1\}_3)_{RX} & \cdots & h_0 \cdot (\{p_n\}_3)_{RX} \\ h_0 & \cdots & h_0 \end{bmatrix} = C \cdot (\widetilde{P})_{RX} \tag{5.7}$$

Por supuesto, es necesario disponer de alguna forma de saber a qué punto tridimensional corresponde cada punto detectado en la imagen, pero esto es una tarea que corresponde al algoritmo de procesado de imágenes que se encarga de localizar dichos puntos (localización de primitivas bidimensionales). Se puede emplear información morfológica, de color, etc… y sólo en última instancia se debería acudir a la opción de probar todas las combinaciones y seleccionar la correcta mediante algún tipo de medida

sobre la coherencia de los resultados. Una técnica común es asociar todos los puntos tridimensionales a marcas de alto contraste sobre una superficie plana en el espacio real, ya que no hay ningún motivo en contra de que los puntos del espacio sean coplanarios (aunque pueden ser necesarias algunas modificaciones, en métodos como el Tsai o en el empleado finalmente en la implementación de este sistema), y así su identificación y su ordenación se simplifican.

Volviendo a lo anterior, ni siquiera sería necesario conocer los parámetros $f, \{t_0\}_1, \{t_0\}_2, \{t_0\}_3, \theta, \varphi$ para llevar a cabo visión estereoscópica, ya que una vez determinada la matriz C asociada a cada cámara, valdría con emplearla en un sistema de ecuaciones indeterminado con un grado de libertad para hallar la recta de posibles puntos tridimensionales correspondientes a un punto imagen. En cualquier caso, se quiera o no, los coeficientes no se pueden resolver directamente por métodos lineales porque forman expresiones no lineales y bastante complejas dentro de cada elemento de la matriz *C*. La propuesta de Duda y Hart es tratar *C* como una matriz 3 x 4 cualquiera, olvidándonos de las expresiones internas que se han hallado, linealizando así el problema a costa de aumentar el número de incógnitas de 6 a 12 (de momento). Se debería por tanto aumentar también la colección de puntos del patrón a 4. El método para resolver una matriz real genérica en función de un conjunto de puntos iniciales y puntos imagen conocidos no se menciona por Duda y Hart ya que, como se verá inmediatamente, hay un detalle que impide que sea usado, pero sí será discutido en otro apartado de esta memoria, para un propósito diferente. El otro inconveniente que se presenta al linealizar la matriz es la imprecisión: al prescindir de las relaciones entre los coeficientes de la matriz, se aumentan los grados de libertad y se permiten estados que no son válidos para la matriz según el modelo desarrollado. Es más, todo el trabajo realizado construyéndolo sería inútil si nos quedamos aquí. Duda y Hart proponen emplear la resolución lineal descrita como primer paso aproximativo, y seguirla de una búsqueda iterativa por gradiente que sí emplee el modelo construido. Por tanto no se trata realmente de un método de resolución lineal del problema, sino de un método de búsqueda por gradiente cuyo punto inicial se resuelve por otro método lineal aproximado.

Sin embargo, hay un dato muy importante a señalar, quizá el que más. Como se comentó antes, la constante de proporcionalidad que permite pasar de coordenadas homogéneas a originales no se conserva a través de la aplicación, ni tampoco se conserva entre imágenes de dos puntos distintos. Esto se ha recalcado en la notación de (5.7), donde se puede ver que la constante es la misma para todos los puntos tridimensionales de $(\widetilde{P})_{RX}$, ya que se han convertido desde las coordenadas tridimensionales (h_0 puede ser igual a 1 por comodidad), pero en cada uno de los puntos imagen de $\widetilde{P}_p$, la cuarta coordenada es distinta. Esto hace que si se dispone solamente del conjunto de *n* puntos tridimensionales y *n* puntos imagen, y estos últimos están en su formato no homogéneo:

$$P_p = \begin{bmatrix} \{p_{p1}\}_1 & \cdots & \{p_{pn}\}_1 \\ \{p_{p1}\}_3 & \cdots & \{p_{pn}\}_3 \end{bmatrix} \tag{5.8}$$

que es el que la cámara ofrece, no pueda emplearse (5.7) directamente, ya que no puede obtenerse $\widetilde{P}_p$ a partir de P_p sin conocer *C*, que es precisamente la incógnita para cuya resolución debe conocerse $\widetilde{P}_p$. La forma de salir de este problema propuesta por Duda y Hart es definir una matriz diagonal:

$$D = \begin{bmatrix} d_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & d_n \end{bmatrix} \tag{5.9}$$

e introducirla en el sistema (5.7) , de forma que cada elemento multiplique un punto imagen en coordenadas homogéneas obvias (con h_0 como cuarta coordenada, con la doble tilde para denotar este tipo de conversión a homogéneas), convirtiéndolo a la versión proporcional que produce la aplicación *C*. Dado que consideramos $h_0 = 1$ por sencillez, resulta que $d_i = h_i$, y el sistema queda como:

$$\widetilde{\widetilde{P}}_p \cdot D = \begin{bmatrix} \{p_{p1}\}_1 & \cdots & \{p_{pn}\}_1 \\ \{p_{p1}\}_3 & \cdots & \{p_{pn}\}_3 \\ 1 & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} h_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & h_n \end{bmatrix} = \widetilde{P}_p = C \cdot (\widetilde{P})_{RX} \tag{5.10}$$

Ahora sí, $\widetilde{\widetilde{P}}_p$ es construible a partir de los puntos imagen obtenidos de una cámara simplemente, como $(\widetilde{P})_{RX}$ lo era ya a partir de los puntos tridimensionales, considerando siempre la cuarta coordenada con valor constante e igual a 1. Sigue tratándose de un sistema de ecuaciones lineales, aunque un poco más complicado. Dado que el número de incógnitas, considerando los coeficientes de la matriz *D,* es *12+n* y el número de ecuaciones es *3n*, resulta que el mínimo número *n* de puntos para resolverlo es 6 en vez de los 4 previstos inicialmente. Duda y Hart proporcionan un método para resolver C y D mediante un sistema sobredeterminado, usando más de 6 puntos, a través de unas ecuaciones que garantizan una solución con mínimo Error Cuadrático Medio mediante el uso de pseudoinversas. Se enuncian a continuación los pasos, aunque como hacen ellos, se omite la demostración. Sean:

$$\begin{aligned} A &= \widetilde{P}^t (\widetilde{P}\widetilde{P}^t)^{-1} \widetilde{P} - I \\ Q &= (\widetilde{\widetilde{P}}_p)^t \widetilde{\widetilde{P}}_p \\ S &= \{s_{i,j}\} / s_{i,j} = a_{i,j} \cdot q_{i,j} \\ d &= \begin{bmatrix} h_1 & \cdots & h_n \end{bmatrix}^t \end{aligned} \tag{5.11}$$

Entonces, se puede hallar d y por consiguiente D resolviendo el sistema lineal homogéneo:

$$S \cdot d = 0 \tag{5.12}$$

y la matriz C mediante

$$C = \tilde{\tilde{P}}_p D \tilde{P}^t (\tilde{P}\tilde{P}^t)^{-1} \tag{5.13}$$

Con las matrices D y C halladas, puede deducirse el valor de los parámetros progresivamente:

$$\begin{aligned} \theta &= \arccos(C_{1,1}) \\ \varphi &= \arccos(C_{3,3}) \\ f &= \frac{\sin(\varphi)}{C_{4,3}} \end{aligned} \tag{5.14}$$

Con los anteriores parámetros hallados, $\{t_0\}_1, \{t_0\}_2, \{t_0\}_3$ se pueden determinar construyendo un sistema de ecuaciones lineales 3 x 3 con la información correspondiente a $C_{1,4}$ $C_{2,4}$ y $C_{3,4}$.

Finalmente, tendremos el vector:

$$\pi = \begin{bmatrix} f & \{t_0\}_1 & \{t_0\}_2 & \{t_0\}_3 & \theta & \varphi \end{bmatrix}$$

Ahora, empleando este vector como punto de partida, se puede realizar una búsqueda por gradiente sobre la función:

$$P_p = C((P)_{RX}, \pi)$$

Esto es más bien un cambio de notación, aunque hay que señalar que ya no es necesario emplear coordenadas homogéneas, ya que C consiste ahora en una función no lineal que depende de los 6 parámetros.

Por último, recordar que como se dijo con anterioridad, el hecho de no considerar el ángulo *roll* no tiene grandes repercusiones en la primera etapa del proceso descrito, pero para la segunda, la de búsqueda por gradiente, sería necesario reescribir las ecuaciones y considerar una búsqueda en 7 dimensiones. En todo caso, la metodología es la misma.

2.2. IMPLEMENTACIONES EXISTENTES

La variedad existente de sistemas de visión estereoscópica para aplicaciones industriales es muy grande, y su recopilación se sale del objetivo de este texto. Además, muchos de ellos emplean métodos de calibrado y localización adaptados a cámaras muy específicas, con coeficientes de distorsión bajos o muy conocidos, y precios muy altos.

Por los mismos motivos que los expuestos en el apartado "Calibrado del sistema", por los cuales no se empleaban métodos de calibrado que consideraran las distorsiones ópticas, tampoco se ha hecho uso de bibliotecas desarrolladas por terceros, ni en general de cualquier implementación de métodos de calibrado o localización existente, que en general consideran estas distorsiones. También como en ese caso, esta decisión será revisada críticamente en el apartado de Conclusiones y mejoras de este proyecto.

Aún así, como descripción superficial del estado del arte, merecen ser destacadas algunas librerías existentes:

"*Open Source Computer Vision Library*". (OpenCV). Librería de código abierto, implementada en C/C++, multiplataforma, que está orientada a la visión por computador en general, pero posee funciones para el calibrado de cámaras y visión estereoscópica. Como Duda y Hart y Tsai, proporciona métodos de calibrado basados en patrones con muchos puntos, y como el segundo, incluye la distorsión radial de la lente en el modelo. *Ver Bibliografía :*[*Impl.1*].

"*Camera Calibration toolbox for Matlab*". Es la implementación en lenguaje Matlab de las rutinas de calibrado de OpenCV. El código es obra de Jean-Yves Bouguet, Ph.D., del *Computer Vision Research Group* (Departamento de Ingeniería Eléctrica, California Institute of Technology). *Ver Bibliografía :*[*Impl.2*].

"*Tsai Camera Calibration Software*". Implementación en lenguaje C del método de Tsai, así como de una extensión del mismo que introduce más parámetros y resulta más pesada computacionalmente. Mantenida por Reg Willson, Ph.D, miembro del "*Machine Vision Group"* (*Jet Propulsion Laboratory*, NASA) *Ver Bibliografía :*[*Impl.3*].

3. DESARROLLO DEL PROYECTO

3.1. CARACTERÍSTICAS DEL SISTEMA

A continuación se valorará las consecuencias sobre el diseño del sistema que tienen los objetivos principales enunciados con anterioridad.

Consecuencias del Objetivo 1. Si se busca un seguimiento en tiempo real, hay que hacer extremadamente ligera la algoritmia del seguimiento del objeto. Especialmente si se considera lo señalado en el Objetivo 3, ya que si no se dispone de un hardware de cómputo dedicado el rendimiento de las operaciones será menor, y éstas deberán compartir el procesador en un SO multiproceso con, al menos, la aplicación que corra por encima de ella y que haga uso de su salida (GUI 3d, programa CAD, aplicación de realidad virtual, …). Marcando como meta orientativa conseguir un periodo de refresco de 10Hz, y considerando un margen para las otras aplicaciones, cada refresco de la localización debería tomar alrededor de 50ms máximo. Para hacer un análisis previo de las medidas que han de tomarse para que esto sea viable, consideremos las dos etapas que componen básicamente cada refresco de la localización: Extracción de las primitivas 2d de las imágenes de cada cámara, y cálculo de la primitiva 3d en función de las primitivas 2d.

Mientras que se puede ver que el tiempo que puede tomarse un procesador típico actual en resolver las operaciones matemáticas involucradas en la segunda etapa (resolución de un sistema lineal de pocas incógnitas) es suficientemente pequeño, lo concerniente a la primera etapa es bastante más oscuro. Reconocer un objeto determinado en una imagen natural puede ser una tarea que requiera mucha carga computacional. Como ilustración, sólo una operación de convolución para buscar la correlación de una máscara de 30x30 píxeles en dos imágenes de baja resolución (320x240) se toma 140 millones de operaciones individuales. Teniendo en cuenta que en condiciones generales en las que no se dispone de información a priori sobre cercanía u orientación del objeto a localizar, suele ser necesario componer esta operación con otras de escalado, deformación o proyección (lo que se asemeja a una transformada de Hough generalizada) se hace absolutamente imposible mantener el tiempo real en un hardware actual de coste medio. Por tanto, es evidente que hay que aligerar esta etapa con algún tipo de "ayuda" que haga la proyección del objeto más evidente dentro de la escena. El objetivo es conseguir que el número de operaciones se acerque lo más posible al número de píxeles en todas las imágenes, es decir, que el algoritmo de exploración haga un único barrido de cada imagen, con los extras mínimos.

La forma más directa de hacer esto es conseguir que la proyección del objeto arroje directamente valores máximos para cierta banda. Opciones más elaboradas como emplear operadores estadísticos locales, como la varianza, para detectar una textura determinada, nos lleva al problema anterior del número de operaciones, ya que para conseguir una caracterización suficientemente significativa, el entorno local debería ser bastante amplio.

Emplear las bandas de color por sí solas es insuficiente, porque no se puede asegurar que en una escena no controlada no haya objetos que arrojen valores similares. En cuanto a la luminancia, si se trata de iluminación pasiva, se puede decir lo mismo. Por lo tanto, la única alternativa para conseguir el objetivo marcado sobre el número de operaciones es emplear iluminación activa. Además, esto tiene la incalculable ventaja de que se puede remarcar la diferencia de luminosidad entre el objeto y el entorno bajando el tiempo de exposición programáticamente, una característica común en la mayoría de las cámaras baratas actuales (ver consecuencias del Objetivo 3 para discusión sobre el tipo de cámaras de vídeo digitales empleadas). La razón de que se remarque el objeto luminoso es que en condiciones habituales, en cuanto el emisor de luz supera cierto umbral de potencia bastante bajo, provoca saturación en el sensor de la cámara, ya que la sensibilidad de esta está adecuada a captar el rango de luminancia de objetos con iluminación pasiva. Esta saturación se mantiene para niveles de exposición menores, que en cambio oscurecen el resto de la escena, compuesta mayoritariamente por objetos con iluminación pasiva. Cuando la exposición baja hasta un punto en el que la fuente luminosa ya no satura el sensor, la diferencia entre la luminancia pico conseguida y la luminancia media del fondo es mucho mayor que para tiempos de exposición adecuados para observar una escena natural. Además, bajar el tiempo de exposición tiene la ventaja colateral de evitar el efecto de integración temporal que se da en movimientos rápidos, y que desemboca en líneas donde debería haber puntos.

Físicamente, los puntos con iluminación activa se pueden conseguir usando LEDs de potencia media. En caso de que se desee omnidireccionalidad, pueden configurarse varios juntos, o usar alguna técnica para dispersar la luz de uno solo. La información de color, también disponible si se emplean LEDs, se puede emplear aquí como información de apoyo en la decisión, y puede ser útil a la hora de abordar el objetivo adicional de la detección de varios puntos simultáneamente, si el uso de la recta epipolar de búsqueda falla por alguna razón (ya se ha comentado esta posibilidad, y en "Consecuencias del Objetivo 2" se discute nuevamente). Es más, debido a que la crominancia de un LED viene dada porque según el compuesto empleado en la unión PN, el espectro de emisión se centra en una longitud de onda específica, y esto lo hace energéticamente más eficiente que una fuente con crominancia basada en filtros que absorben las longitudes de onda no deseadas de una fuente de espectro más plano. Este ahorro energético es útil si se pretende que el dispositivo señalador (Objetivo 2) funcione con batería. Los resultados experimentales indican que también ofrece una componente de hue más constante, lo cual seguramente se deba al mismo motivo.

El uso de LED's como objetos localizables va en la dirección contraria del objetivo 3, ya que es un dispositivo que el usuario posiblemente ha de adquirir, pero es un mal menor frente a la adquisición de un hardware que permita la detección de patrones menos obvios en tiempo real. Además, dispositivos LED de tales características existen en abundancia construidos para otros propósitos, y la omnidireccionalidad, si se hace necesaria, se puede conseguir con pequeñas modificaciones. En última instancia, el usuario podría recurrir a fuentes blancas filtradas, aunque provocaran tasas de error mayores.

Consecuencias del Objetivo 2. El hecho de que el sistema se use para localizar un apuntador o un objeto virtual en general establece el rango de distancias que deberá cubrir: las que encierren un espacio suficiente para incluir cualquier movimiento de brazos del usuario, y opcionalmente un desplazamiento. Por otro lado, tiene una consecuencia muy importante: no será necesario que el sistema demuestre una precisión tan elevada como, por ejemplo, pueden tener sistemas de visión estereoscópica orientados a procesos industriales. Las condiciones que sí que han de cumplirse para que el sistema sirva a tal propósito son las siguientes:

- Que haya una relación uno a uno entre los puntos del espacio real y los puntos del espacio virtual, tanto a corto como a largo plazo. Si el objeto localizable sólo fuese a emplearse como dispositivo señalador, este requisito podría saltarse, ya que muchos dispositivos señaladores de dos dimensiones, como el ratón, no la cumplen ni a corto plazo (por el efecto de aceleración que emplea la mayoría) ni a largo (por la posibilidad de levantar el ratón de la superficie y depositarlo en otro punto de ella sin que el puntero virtual se desplace). Sin embargo, como el objetivo es que este sistema sirva también para localizar la posición absoluta del objeto localizable como representante de un objeto dentro de un contexto virtual, esta condición debe cumplirse.

- Si *(x,y,z)* es un punto del espacio localizable, y *L(x,y,z)* es el punto del espacio virtual donde la máquina sitúa a *(x,y,z)* según la condición anterior, sea la función error cuadrático $(\Re^3 \rightarrow \Re)$ $E(x,y,z) = ((x,y,z)\text{-}L(x,y,z))^2$. La segunda condición es que la función *E*, aunque pueda alcanzar magnitudes mayores que en otras aplicaciones, esté acotada por un valor no demasiado alto E_{MAX} , así como el módulo de su gradiente por $E'_{MAX.}$ (los valores de estas cotas dependerán de la exigencia de la aplicación). La condición concerniente al gradiente indica que no habrá variaciones bruscas en E a lo largo de $\Re^3$, que desembocarían en deformaciones de la trayectoria perceptualmente distinguibles.

Cumpliéndose estas condiciones, el resultado puede describirse en lenguaje informal como que el espacio virtual puede quedar deformado con respecto al real, pero con una deformación homogénea, similar a un 'estiramiento' leve, y estable en el tiempo.

Dado que las distorsiones introducidas por la lente introducen errores en el espacio que cumplen lo descrito, es posible ignorarlas sin perjuicio de los objetivos propuestos. De esta forma, podrá emplearse exclusivamente el modelo pinhole, lo que se traducirá en una algoritmia más sencilla a la hora de implementar, y más ligera en tiempo de ejecución.

Sin embargo, hay que notar que la distorsión sobre las imágenes bidimensionales no sólo tiene repercusiones en la localización tridimensional, sino también en la localización de las primitivas bidimensionales sobre las propias imágenes. La primitiva bidimensional puede quedar fuera de la recta de búsqueda que le correspondería dentro del modelo epipolar, como se explicó en el apartado que introduce este tipo de geometría. También ahí, se llegó a la conclusión de que con distorsiones que no pudieran conocerse a priori y compensarse, la recta de búsqueda (línea epipolar) necesitaría aceptar anchos muy grandes que mermarían la ventaja de la eficiencia de búsqueda, convirtiéndose más bien en franjas cuya anchura además sería difícil de determinar a priori (al desconocer las distorsiones, habría que emplear algún método de

ensayo y error). Si por encima son varios los puntos a localizar en el espacio, que es uno de los objetivos opcionales propuestos, podrían darse grandes ineficiencias, ya que las franjas de búsqueda se solaparían unas con otras, llegando posiblemente a consumir más operaciones que un barrido único. Dado que las cámaras a emplear son de bajo coste (Objetivo 3, más en detalle en Consencuencias del Objetivo 3), es de esperar que las distorsiones ópticas sean grandes, y sus parámetros no sean proporcionados por el fabricante. Dado lo difícil que sería, para cámaras de bajo coste arbitrarias, modelar todas las distorsiones involucradas en el problema descrito para la recta de búsqueda, se ha optado por no hacer uso de las rectas de búsqueda, y explorar todas las imágenes con un barrido único y ordenado. Esto provoca una algoritmia más sencilla (en lenguajes de bajo nivel, además, suele ser eficiente la exploración ordenada de un array, por el efecto de localidad), y no necesariamente menos eficiente, por los motivos expuestos.

Consecuencias del Objetivo 3. Renunciar a hardware altamente especializado, tanto en materia de cámaras como de plataforma de cómputo, tiene repercusiones muy importantes.

-Como ya se ha comentado, las cámaras empleadas serán extremadamente económicas, y por tanto no métricas. Es decir, carecerán de las ventajas propias de aquellas empleadas en muchos sistemas de visión artificial: conocimiento a priori de parámetros intrínsecos (distancia focal y distorsiones ópticas principalmente) y minimización de estas últimas. Estos parámetros, por tanto, deberán ser estimados o ignorados por el propio sistema. Aunque, como se ha dicho en el apartado anterior, no se van a considerar las distorsiones ópticas. Las aberraciones y distorsiones en estos dispositivos son mayores que en otros, pero se ve recompensado por el bajo coste, la disponibilidad y la facilidad de uso para un usuario común.

-La capacidad de cómputo será menor que si se dispusiera de hardware especializado. Las consecuencias de esto ya se han discutido junto con el objetivo 1. Por otro lado, también puede haber mayores retardos que en cámaras diseñadas para un propósito específico, debido a los distintos búferes y buses de propósito general por los que pasarán los flujos de datos de las imágenes.

Consecuencias del Objetivo 4. Como se ha explicado anteriormente, el sistema debe conocer la posición y orientación relativa de las cámaras, así como sus distancias focales. La tarea de colocar las cámaras en una posición determinada y orientación determinadas requiere cierta instrumentación precisa, ya que una mera colocación cualitativa, sobre todo en cuanto a orientación respecta, provocaría fácilmente errores demasiado grandes. Pero esto va en contra del objetivo de no emplear instrumentación específica. Además, también contradice el que busca maximizar la autonomía del sistema. Por otro lado, muchos fabricantes de cámaras de bajo coste ni siquiera publican la distancia focal (haría falta, en cualquier caso, conocer la distancia focal y el tamaño del sensor). Todo esto desemboca en una necesidad que modifica profundamente el alcance del proyecto: el sistema debe calibrarse automáticamente. Lo que es lo mismo, ha de extraer los parámetros de las cámaras, tanto extrínsecos (posición y orientación relativas) como intrínsecos (distancia focal). El resto de parámetros intrínsecos, como ya se ha comentado, no se considerarán.

El calibrado se hará obteniendo por cada cámara una imagen de un patrón complejo, y gracias a la relación entre la geometría del patrón tridimensional y su proyección cónica, se podrá inferir la distancia focal de la cámara y la situación del patrón con respecto a ella. En concreto, el patrón consistirá en un conjunto de puntos en el espacio tridimensional, cuyas proyecciones deberán reconocerse en las imágenes obtenidas mediante una técnica de procesado adecuadas, y la información que se empleará para la resolución matemática del problema serán las distancias relativas entre los puntos tridimensionales del espacio, y las distancias entre sus proyecciones en el plano. El reconocimiento de estas proyecciones en el plano no deberá ser en tiempo real, ya que es una tarea que se hará una sola vez tras la sustitución o cambio de posición de alguna de las cámaras, y por lo tanto no será necesario emplear puntos con iluminación activa y fotogramas subexpuestos, bastando el uso de un patrón de dimensiones exactas conocidas, y con un suficiente número de puntos que sean fácilmente reconocibles en la imagen natural.

Posteriormente, con la orientación relativa entre el patrón de calibrado y cada cámara, podrá obtenerse la orientación relativa entre las cámaras, con lo que ya se tendrá toda la información necesaria para emplear visión estereoscópica. Los métodos empleados para todos estos procesos se describen más adelante.

3.2. DESCRIPCIÓN DEL SISTEMA

El sistema consistirá, por tanto, en los siguientes módulos.

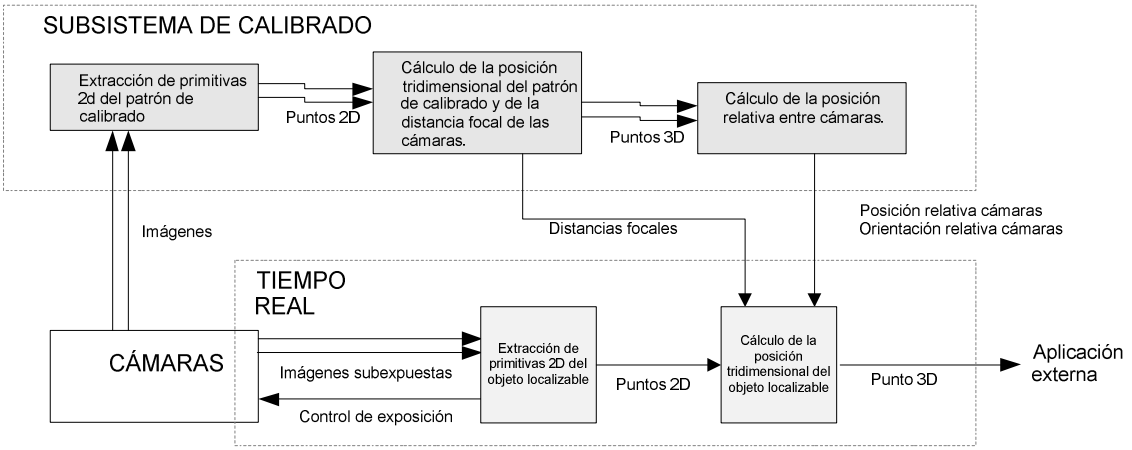


Figura 8.1

A continuación se hará una descripción detallada del sistema, dividida por módulos. No se seguirá el orden natural de ejecución en la máquina, sino el más adecuado para explicar uno en función de los resultados de los anteriores. Como los resultados son a menudo unos requisitos de entrada, se ha elegido un orden de exposición diferente al de ejecución.

SUBSISTEMA DE CALIBRADO

- Fase 2 de calibrado: Cálculo de la posición tridimensional del patrón de calibrado y de la distancia focal de las cámaras
- Fase 3 de calibrado: Cálculo de la posición relativa entre cámaras
- Fase 1 de calibrado : Extracción de las primitivas 2d del patrón de calibración

SUBSISTEMA DE LOCALIZACIÓN EN TIEMPO REAL

- Fase 2 de localización en tiempo real : Cálculo de la posición tridimensional del punto
- Fase 1 de localización en tiempo real: Extracción de las primitivas 2d del objeto localizable

Fase 2 de calibrado: Cálculo de la posición tridimensional del patrón de calibrado y de la distancia focal de las cámaras

En el estudio previo se ha descrito un método para extraer los parámetros intrínsecos y extrínsecos de una cámara en modelo pinhole a través de la observación de un patrón de *n* puntos, propuesto en Duda y Hart (1973). Este método intenta, mediante el uso de coordenadas homogéneas, tratar la proyección cónica como una aplicación lineal, y expresada matricialmente en composición con aplicaciones naturalmente lineales como la traslación y la rotación en el espacio, modelar el proceso de captación de una cámara pinhole con parámetros intrínsecos y extrínsecos genéricos como una aplicación lineal. Sin embargo, se ve que las coordenadas homogéneas no logran eliminar realmente la no linealidad del proceso a la hora de despejar los parámetros de búsqueda, y finalmente hay que realizar una segunda etapa de búsqueda no lineal por gradiente para llegar a un resultado preciso. Sin embargo, para poder realizar la primera etapa de resolución lineal aproximada, ha sido necesario aumentar el número de incógnitas, y por ende el de ecuaciones y el de puntos en el patrón de calibrado. Además, la versión concreta propuesta por Duda y Hart considera que la cámara nunca va a estar inclinada lateralmente, lo cual en un sistema como el que aquí se plantea, consistente en cámaras de bajo coste sin instrumentación para una colocación precisa, se convierte en una simplificación poco realista.

Por estos motivos, se ha empleado un método más directo para el calibrado del sistema, que desde el principio expresa el problema como un sistema de ecuaciones no lineales, y lo resuelve por un método iterativo. La justificación consiste en que si el calibrador está abocado a emplear un método de este tipo finalmente, es preferible hacerlo directamente y así requerir un patrón con un número de puntos mínimo, es decir, ajustado al número real de grados de libertad del sistema. La posible desventaja de prescindir de la primera etapa es que la carencia de una solución aproximada como punto de partida para la búsqueda iterativa puede hacer que la convergencia no sea adecuada, y por tanto abrir la necesidad de repetir el algoritmo con un gran número de puntos de partida distintos y validar las soluciones con alguna medida de coherencia basada en la sobredeterminación del sistema (como se verá más adelante, empleando un objeto rígido como patrón de calibrado este grado de sobredeterminación se presenta por sí solo). Sin embargo, con el método implementado, los resultados experimentales obtenidos indican que el retraso temporal introducido es casi imperceptible si se emplean baterías de puntos iniciales adecuadas y un procesador actual estándar.

Por motivos de rigor, hay que señalar desde ahora que el método que se ha empleado finalmente, aunque iterativo y diseñado para la resolución de sistemas no lineales, no es de búsqueda por gradiente como se propone en Duda y Hart para la segunda fase. Se entrará en detalles más adelante.

En el enfoque empleado para la calibración, se ha abandonado la notación matricial para describir la posición y orientación de la cámara con respecto a una referencia dada, sea o no ajustada al patrón. En el enfoque de Duda y Hart se empleaba un sistema de referencia ajeno tanto a la geometría de la cámara como la del patrón. En este caso se ha optado por describir los puntos del patrón en función del sistema de referencia asociado a la cámara, de modo que una vez resuelto el sistema, la traslación y la rotación quedan implícitas como las que convierten entre el sistema de referencia empleado, que corresponde al de la cámara, y uno asociado a la posición y orientación del patrón. Así,

posteriormente se puede obtener la relación entre posición y orientación de dos cámaras, ahora ya como formato aplicación lineal en formato matricial, componiendo las aplicaciones de conversión del sistema de una cámara al sistema del patrón, y del sistema del patrón al sistema de otra cámara.

La relación entre una cámara y su sistema de referencia asociado también cambia con respecto a los apartados anteriores, y la aquí empleada es también la que se usará en adelante. Difiere con la que se empleó en el estudio previo (siguiendo el planteamiento de Duda y Hart) en que el centro del sistema de referencia asociado a una cámara está en su objetivo, y el eje óptico coincide con el Z en vez de con el Y. La justificación del primer cambio es que de la otra forma, al cambiar la distancia focal se cambiaba la posición del objetivo y por tanto también la de la cámara. La posición de la cámara está asociada a la de su objetivo y no a la del plano imagen porque sólo si el objetivo se mantiene inmóvil al variar la distancia focal resulta que el resultado es un cambio de escala en la imagen. De la otra forma, se compone este efecto con el de un desplazamiento de la cámara en la dirección del eje óptico. Aunque en la mayoría de cámaras físicas es ciertamente el sensor el que se mantiene inmóvil y es el objetivo el que se desplaza al cambiar f, la diferencia es generalmente despreciable por ser las dimensiones del sensor y de la distancia focal muy pequeñas en comparación con la distancia a los puntos proyectados. Pero dado que en nuestro modelo de cámara se emplean unas dimensiones grandes (conservando la proporción entre sensor y distancia focal, como ya se explicó antes) las diferencias sí podrían ser notables para distintas distancias focales típicas, y conviene emplear la convención elegida. De esta forma, la traslación T que forma parte de M también es independiente del valor de f. Al ser el objetivo el origen de coordenadas, también se simplifican ciertas ecuaciones. El segundo cambio, consistente en emplear el plano XY para el plano imagen y el eje Z para el eje óptico en la cámara canónica, permite una mayor claridad al hacer que las coordenadas asociadas a la imagen bidimensional sean contiguas en las expresiones matriciales, y las coordenadas x e y de la imagen coincidan con las del punto proyectado expresado en el sistema de referencia asociado a su cámara (de la otra forma la y equivalía a la z y añadía cierta confusión). En la implementación realizada del sistema, cuyos detalles se describirán más adelante, también se ha empleado esta notación. En cualquier caso, ninguno de estos cambios modifica los resultados y conclusiones obtenidos.

Con todo lo dicho, el escenario para la proyección de dos puntos en una cámara, empleando su sistema de referencia asociado, es el siguiente:

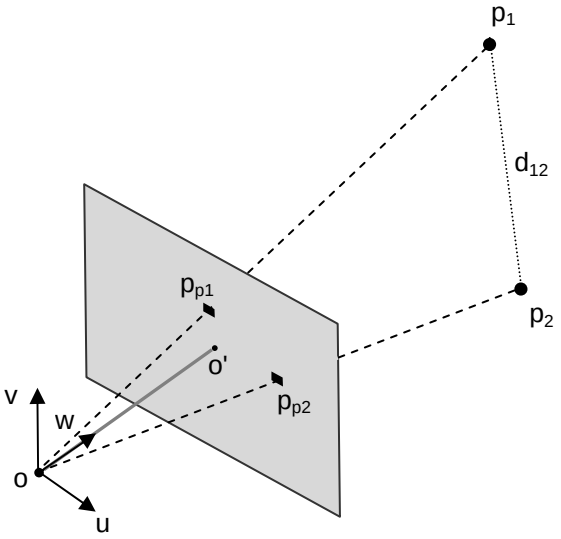


Figura 9.1

Con las siguientes definiciones:

$$\begin{aligned} o &= (0,0,0) \\ u &= (1,0,0) \\ v &= (0,1,0) \\ w &= (0,0,1) \\ o' &= (0,0,f) \\ d_{i,j} &= \left\| p_i - p_j \right\| \end{aligned} \tag{9.1}$$

Gracias al nuevo sistema de referencia, se puede declarar de forma sencilla una relación de proporcionalidad entre cada punto y su proyectado, de la siguiente forma:

$$p_i = \lambda_i p_{pi} \tag{9.2}$$

Dado que el objetivo por el momento es determinar f y la posición de los n puntos del patrón en función del sistema de referencia de la cámara, es decir $\{p_i / i \in [1,n]\}$, y considerando que conocemos las dos primeras coordenadas de v_{pi} por ser las de las primitivas bidimensionales extraídas en la fase 1 de calibrado, la ecuación (9.2) lleva a que la resolución del problema equivale a hallar f y $\{\lambda_i / i \in [1,n]\}$, con el número n de puntos que sea necesario. Se supone que no hay oclusión de ningún tipo y todos los puntos del patrón son visibles, pero la elección de una geometría adecuada para el patrón, de forma que se asegure dicha condición, corresponde nuevamente a la discusión de la fase 1 del calibrado, ya que habrá más variables en juego, como la facilidad para localizar las primitivas bidimensionales en las imágenes. Por tanto, aquí sólo se discute el número de puntos necesarios.

La información de la que se dispone para la resolución es la distancia entre las primitivas tridimensionales: $d_{i,j}$. Por cada par no ordenado de primitivas tridimensionales, se puede realizar el siguiente desarrollo:

$$d_{i,j} = \left\| p_i - p_j \right\| \overset{(9.2)}{=} \left\| \lambda_i p_{pi} - \lambda_j p_{pj} \right\| \overset{(p_{pk})_3 = f}{=} \sqrt{(\lambda_i (p_{pi})_1 - \lambda_j (p_{pj})_1)^2 + (\lambda_i (p_{pi})_2 - \lambda_j (p_{pj})_2)^2 + (\lambda_i f - \lambda_j f)^2} \tag{9.3}$$

Considerando patrones rígidos, tenemos que para dos puntos hay tres incógnitas $(\lambda_1, \lambda_2, f)$ y una ecuación con término independiente $d_{1,2}$. Tres puntos sería el mínimo necesario para determinar un movimiento rígido, que es el objetivo final (los cómos se explican el Fase 3 de calibrado). Sin embargo, hay cuatro incógnitas $(\lambda_1, \lambda_2, \lambda_3, f)$ y tres ecuaciones con términos independientes $(d_{1,2}, d_{1,3}, d_{2,3})$, lo cual sigue sin proporcionar un sistema determinado. Para cuatro puntos, son 5 incógnitas $(\lambda_1, \lambda_2, \lambda_3, \lambda_4, f)$ y $C_{4,2} = \frac{4!}{2! \cdot 2!} = 6$ ecuaciones. En este caso el sistema queda resuelto con un grado de sobredeterminación, pero dado que es un sistema no lineal y puede haber un número finito de soluciones correctas aunque el número de ecuaciones sea igual al de incógnitas, la ecuación sobrante se empleará como medida de coherencia para determinar si la solución encontrada es la adecuada o no. Hay que notar, eso sí, que hay ciertas soluciones que sin ser las adecuadas desde la física del problema, son fácilmente convertibles, como la $(-\lambda_1, -\lambda_2, -\lambda_3, -\lambda_4, f)$ y la $(-\lambda_1, -\lambda_2, -\lambda_3, -\lambda_4, -f)$. La implementación desarrollada contiene el mecanismo necesario para detectar estos casos y convertirlos a la solución adecuada.

Por tanto, el sistema resultante se puede reescribir como:

$$\begin{gathered} (\lambda_i (p_{pi})_1 - \lambda_j (p_{pj})_1)^2 + (\lambda_i (p_{pi})_2 - \lambda_j (p_{pj})_2)^2 + (\lambda_i f - \lambda_j f)^2 - {d_{i,j}}^2 = 0 \\ \forall i, j \in [1,4] \end{gathered} \tag{9.4}$$

Desarrollando los cuadrados, empleando una notación más específica en los subíndices, y omitiendo la ecuación sobrante, queda en:

$$\begin{gathered} (\lambda_i (p_{pi})_1)^2 + (\lambda_j (p_{pj})_1)^2 - \lambda_i \lambda_j (p_{pi})_1 (p_{pj})_1 + \\ (\lambda_i (p_{pi})_2)^2 + (\lambda_j (p_{pj})_2)^2 - \lambda_i \lambda_j (p_{pi})_2 (p_{pj})_2 + \\ (\lambda_i f)^2 + (\lambda_j f)^2 - \lambda_i \lambda_j f^2 - {d_{i,j}}^2 = 0 \\ \forall (i, j) \in \{(1,2), (1,3), (1,4), (2,3), (2,4)\} \end{gathered} \tag{9.5}$$

El sistema es claramente no lineal. Si se considerara *f* conocida, cada ecuación sería una forma cuadrática con matriz y resultado constante, y por las propiedades de esas matrices el sistema podría interpretarse geométricamente como la intersección de cuatro

cilindros en $\Re^4$, cada uno paralelo y centrado en un eje canónico, pero de corte elipsoidal y con excentricidad y orientación arbitrarias. Aún con esta simplificación, no ha sido posible encontrar una solución analítica, y menos aún en el caso en el que *f* es una cuarta incógnita que entra dentro de la matriz cuadrática, llegando cada ecuación a grado 4. En cualquier caso, tras la lectura de otros métodos de resolución también basados en el modelo Pinhole, como el expuesto por Duda y Hart, era esperable no encontrar una solución directa. Por tanto, se ha acudido a métodos iterativos de resolución de sistemas no lineales.

Análisis de métodos iterativos para la resolución de sistemas de ecuaciones no lineales

Existe multitud de estos métodos, y la teoría subyacente, especialmente en lo que respecta a condiciones y velocidad de convergencia, es muy compleja, quedando fuera del alcance de este texto una discusión en gran profundidad. Se van a plantear tres métodos de resolución presentes en la mayoría de la literatura, y en base a sus propiedades más visibles, se va a justificar la implementación experimental de uno de ellos. Un funcionamiento satisfactorio para los objetivos del sistema se considerará justificación suficiente para su uso definitivo. Aún así, se describirá cada método de forma que se facilite una posible implementación alternativa.

Antes de entrar en cada descripción individual, conviene hacer ciertas anotaciones que serán válidas para todos los casos. Se tratará un sistema general de *n* ecuaciones no lineales con *n* incógnitas, porque aunque al que se ha llegado en este caso es 5x5, como se verá en apartados posteriores, se va a hacer uso también de una versión con distintas dimensiones. Además, así el planteamiento es más general. En la introducción de cada método se omitirá mención alguna a la ecuación extra, por no ser aplicable a ninguno de los métodos en su versión teórica. En la discusión sobre qué método emplear, sin embargo, sí se mencionará esta ecuación extra. Una consecuencia de no emplear la ecuación extra es que, como se dijo anteriormente, el sistema con tantas ecuaciones como incógnitas, por su naturaleza no lineal, podrá tener un número finito de soluciones, y el método que resuelva el sistema *n x n* podrá converger a cualquiera de ellas como solución válida, aunque sólo haya una que satisfaga la ecuación extra y sea por tanto la solución del problema que se trata. De hecho, los resultados muestran que efectivamente, es fácil que el sistema converja a una falsa solución de este tipo. Sin embargo, por sencillez de notación, en la siguiente exposición de cada método, se hablará en ocasiones de la solución del sistema en singular, y se entrará en el posible tratamiento del problema de la falsa convergencia más adelante.

En general, todos los métodos resolverán el sistema tratando cada ecuación en (9.5) como una función $\Re^n \to \Re$ igualada a cero, centrándose en encontrar raíces comunes para las funciones derivadas de las ecuaciones, y no en encontrar la intersección de las variedades directamente dadas por las ecuaciones mostradas, residentes en $\Re^n$. Cada función será etiquetada como $f_i(s)$, con $i \in [1,n]$ y $s \in \Re^n$ (en el caso de funciones $\Re \to \Re$, se empleará la notación clásica $f(x)$). El sistema global podrá escribirse como $F(s) = 0$, siendo $F : \Re^n \to \Re^n / F(s) = (f_1(s), f_1(s), ..., f_n(s))$. A cada $f_i(s)$ se le llamará función coordenada de $F(s)$.

Aunque se mencionarán, no se hará especial hincapié en los requisitos de continuidad y derivabilidad en cualquier orden de las funciones coordenadas, porque en el caso que se pretende aplicar son polinómicas, y por tanto tienen aseguradas esas condiciones.

Conviene definir ahora el concepto de orden de convergencia: si un método iterativo inicializado en el punto s_0 provoca una sucesión $s_1, s_2, s_3\ s_4, \ldots$ que converge a $\hat{s}$, se dice que converge con orden q si :

$$\exists M > 0 \quad / \quad \frac{\left|s_{k+1} - \hat{s}\right|}{\left|s_k - \hat{s}\right|^q} \leq M \qquad \forall k \in N \tag{9.6}$$

Dado que en una sucesión convergente se dará, por lo menos a partir de determinado *k*, que *M<1*, es intuitivo que a mayor orden de convergencia, mayor velocidad de convergencia. Por ejemplo, si *q=2* el error en una iteración será siempre del orden del cuadrado de la anterior, y si la anterior aproximaba la solución con *n* decimales, la siguiente lo hará, al menos, con *2m* decimales.

Dentro de cada orden de convergencia, existen subcategorías en función del valor del límite del cociente de la ecuación (9.6) cuando $k \to \infty$. Por ello, una convergencia de orden 2 puede ser cuadrática o supercuadrática en función de este valor. En este texto se omitirán referencias a esta distinción, por establecer diferencias de menor categoría a las del orden de convergencia, y por no estar claramente especificada en la explicación de algunos métodos en gran parte de la literatura.

En general, en la implementación de métodos iterativos donde se esperan soluciones en un intervalo acotado y de valores no extremos (si no fuese así podrían emplearse medidas de error relativo), la condición de parada que se emplea es:

$$\left|s_k - s_{k+1}\right| < errorMaximoPermitido \quad \text{OR} \quad k > numMaxIteraciones \tag{9.6.bis}$$

Las dos constantes han de elegirse adecuadamente en función de los requisitos de tiempo y precisión. El primer operador de la disyunción se hace verdadero cuando se encuentra una solución aceptable, mientras que el segundo lo hace cuando no es así.

Método del punto fijo

El método del punto fijo para sistemas no lineales es una generalización del método homónimo para la resolución de ecuaciones no lineales. También es llamado por algunos autores método de iteración funcional, de sustitución sucesiva, o método de aproximaciones sucesivas de Picard.

En el caso de resolución de ecuaciones, aprovecha el hecho de que para funciones $\Re \to \Re$ con un punto fijo ($g(\hat{x}) = \hat{x}$), bajo ciertas condiciones ($g(x)$ continua en $[a,b]$, diferenciable en (a,b), $g'(x) \leq k \leq 1\ \forall x \in [a,b]$, $x_0, \hat{x} \in [a,b]$), la sucesión dada por $x_{k+1} = g(x_k)$ converge a $\hat{x}$. Además, la velocidad de convergencia será mayor cuanto menor sea k, aunque siempre serña de orden 1. Por tanto, si se tiene una ecuación $f(x) = 0$, se puede resolver si se encuentra una función $g(x)$ que tenga por

puntos fijos las raíces de $f(x)$. Si se asume que $f(x)$ es continua y diferenciable en un entorno de la raíz, esto se puede conseguir construyendo $g(x)$ de la siguiente forma:

$$g(x) = x - \alpha \cdot f(x) \tag{9.7}$$

Como se ve, la elección de $g(x)$ no es única, y el parámetro α apropiado dependerá de $f'(x)$, para que se cumplan las condiciones especificadas. Cuando no se cumplen las condiciones, el método fácilmente diverge a valores arbitrariamente grandes.

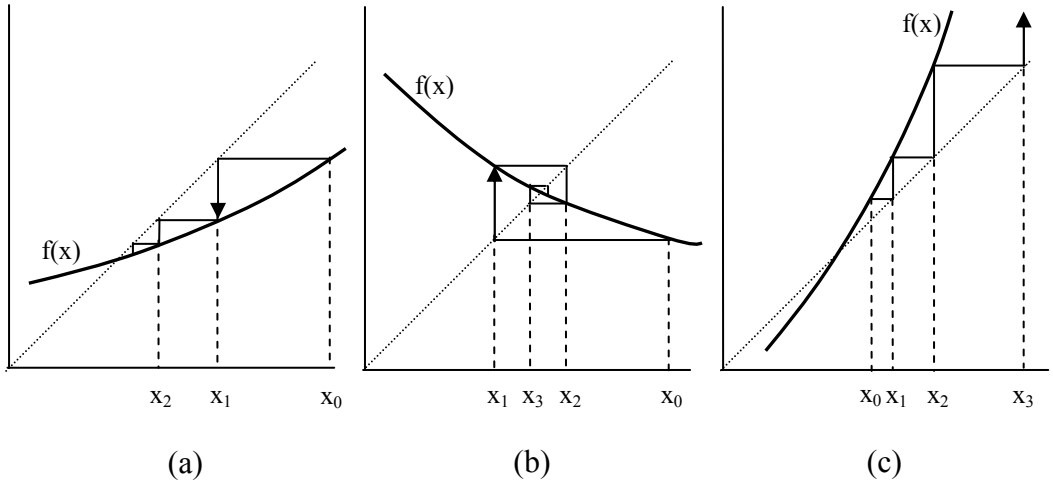


Figura 9.2

La recta *x=y* ayuda a visualizar el proceso en el que la salida de la función cuya raíz se aproxima es la entrada en la siguiente iteración.

(a) Caso convergente con $0<f'(x)<1$.

(b) Caso convergente con $-1<f'(x)<0$.

(c) Caso divergente con $1<f'(x)$.

En el método para la resolución de sistemas, en cada ecuación $f_i(s) = 0$ debe despejarse un componente de s, definiéndose así un número equivalente de funciones $g_i(s) / g_i(\hat{s}) = \{\hat{s}\}_i \Leftrightarrow f_i(\hat{s}) = 0$. Estas funciones $g_i : \Re^n \to \Re$ forman, como funciones coordenadas, una nueva función $G : \Re^n \to \Re^n$ que tiene $\hat{s}$ por punto fijo: $G(\hat{s}) = \hat{s}$. Se habla de punto fijo en singular por no complicar la notación, pero en general *F* puede tener múltiples raíces y *G* múltiples puntos fijos. La convergencia a uno u otro dependerá de la posición relativa del punto inicial de búsqueda, s_0 (aunque se puede ver por un ejemplo que no es necesariamente un criterio de mínima distancia). Una vez se obtiene *G*, se puede aplicar un teorema que expone unas condiciones semejantes a las de la resolución de una ecuación.

Teorema del punto fijo

Sea $D = \{(x_1, x_2, ..., x_n) / a_i \le x_i \le b_i \ \ \forall i \in [1, n]\} \subset \Re^n$ con $\{a_i, b_i\} \subset \Re$ conjunto de constantes.

Sea $\hat{s} \in D$ un punto fijo de G único en D.

Si $G : \Re^n \to \Re^n$ cumple $G(D) \subset D$, es continua y derivable en *D*, y se cumple que:

$$\left| \frac{\partial g_i(s)}{\partial \{s\}_j} \right| \le \frac{K}{n} \le \frac{1}{n} \quad \forall s \in D \quad 1 \le i \le n \quad 1 \le j \le n$$

entonces, si se define $s_0 \in D$, la sucesión dada por $s_{k+1} = G(s_k)$ converge al punto fijo $\hat{s}$.

Además, como antes, la velocidad de convergencia es mayor cuanto menor alcanza a ser la constante *K*. Sin embargo, al ser siempre el orden de convergencia igual a 1, esta será más lenta que en un método con orden 2.

El teorema anterior está expuesto en Burden y Douglas (2002). En Mathews y Fink (1999) la condición $\left| \frac{\partial g_i(s)}{\partial \{s\}_j} \right| \le \frac{K}{n} \le \frac{1}{n}$ se substituye por $\sum_{j=1}^{n} \left| \frac{\partial g_i(s)}{\partial \{s\}_j} \right| \le K \le 1$, que es más débil.

Otro teorema asegura la existencia del punto fijo en *G* sólo con las condiciones enumeradas hasta la continuidad (inclusive), pero no es necesario considerarlo en la aplicación presente, ya que se supone *G* construida a partir de *F* mediante un proceso que asegura la existencia de un punto fijo en *G* por cada raíz en *F*.

Sin embargo, es este proceso, hasta ahora asumido como caja negra, el punto débil del método del punto fijo. En algunos casos basta con resolver algebraicamente $f_i(s)$ para $\{s\}_i$. En otros no es posible tal despeje, o lo es pero los resultados pueden no ser satisfactorios por no cumplir *G* las condiciones expuestas. Cuando la condición de cota de las derivadas no se cumple, es frecuente que los valores de la iteración diverjan con valores cada vez mayores, hasta que desborde el máximo impuesto por el sistema de representación numérica de la máquina. Tanto Burden y Douglas como Chapra concluyen que el método del punto fijo empleado de esta forma es una técnica limitada y que rara vez tiene éxito.

Entonces puede acudirse a la idea de la ecuación (9.7) y extenderla al caso multidimensional, donde una matriz $A \in \mathrm{M}_{nxn}(\Re)$ hace el papel de α .

$$G(s) = s - A \cdot F(s) \tag{9.8}$$

Pero la elección de *A* sigue sin ser obvia para que se cumplan las condiciones expuestas sobre $G(s)$. Es por todo esto por lo que se ha considerado recomendable investigar otros métodos, como el de Newton-Raphson, que se describe a continuación. Sin embargo, al menos una fuente (Burden y Douglas) introduce Newton-Raphson como un caso particular del método del punto fijo, en el que se emplea la ecuación (9.8) eligiendo $A = J^{-1}(F)$. Esta correspondencia existe también para el caso de resolución de ecuaciones. El hecho de que α y A se transformen en $1/f'(x)$ y $J^{-1}(F)$ respectivamente generaliza más que particulariza la propuesta de las ecuaciones (9.7) y (9.8), donde α y A son factores lineales independientes de *x* o *s*, pero sigue siendo válido como forma de obtener *g* o *G* a partir de *f* o *F*. No obstante, dado que no se ofrece un análisis (ni se menciona su existencia) de en qué condiciones dicha elección de *A* implica la satisfacción de las condiciones expuestas para *G*, se ha elegido una introducción basada en la aproximación de las funciones coordenadas por series de Taylor de grado 1 (aproximación lineal), que es más intuitiva. Sin embargo, podrá verse al final que, efectivamente, las expresiones coinciden.

Método de Newton-Raphson

El método de Newton-Raphson para sistemas de ecuaciones no lineales se deriva nuevamente del método de Newton para ecuaciones no lineales. En la modalidad de resolución de una ecuación de la forma $f(x) = 0$, con $f : \Re \to \Re$, el método aproxima sucesivamente una raíz de f calculando, en cada iteración, la raíz de la aproximación por series de Taylor de grado 1 de la función. Ésta viene a ser la función lineal que aproxima mejor la no lineal $f(x)$, en el punto de la iteración actual x_k, construida con los valores $f(x_k)$ y $f'(x_k)$:

$$l_{xk}(x) = f'(x_k)(x - x_k) + f(x_k) \tag{9.9}$$

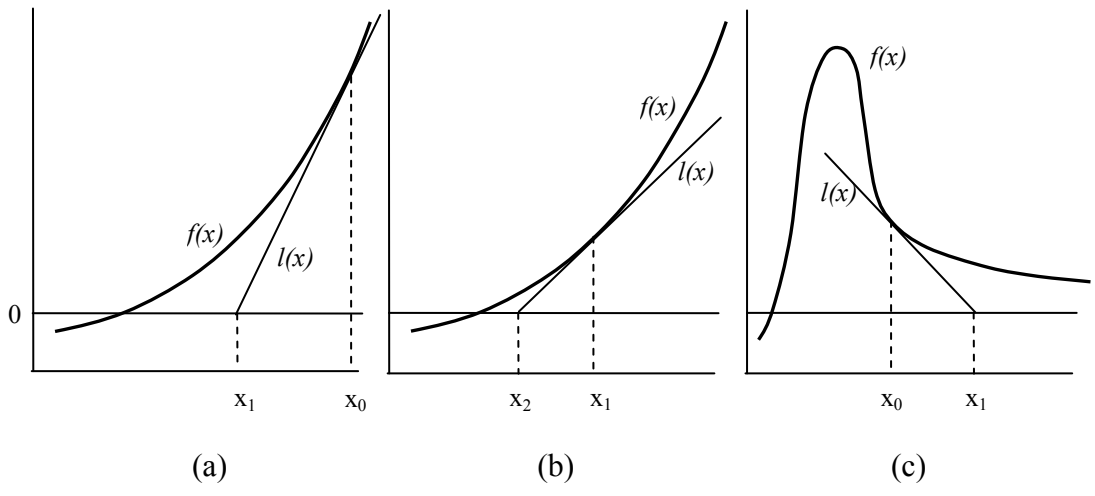


Figura 9.3

En (a) y (b) se pueden apreciar dos iteraciones consecutivas para un caso convergente. El (c), si f(x) tiene una asíntota horizontal en $+\infty$, el método divergirá.

Resolviendo el caso $l(x_{k+1}) = 0$ se obtiene la ecuación generadora del método iterativo:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{9.10}$$

Puede apreciarse la equivalencia mencionada al final de la explicación del método de punto fijo.

El caso multidimensional es análogo al unidimensional. Como hasta ahora, el sistema $F(s) = 0$, $F : \Re^n \to \Re^n$, se puede descomponer en $f_i(s) = 0$ $f_i : \Re^n \to \Re$ $\forall i \in [1, n]$. Para cada función coordenada $f_k(s)$ en el punto s_k se puede construir una aproximación lineal empleando el valor del gradiente en ese punto:

$$\begin{gathered} \nabla f_i(s_k) = \left(\frac{\partial f\ (s_k)}{\partial \{s\}_1}, \frac{\partial f_2(s_k)}{\partial \{s\}_2}, \ldots, \frac{\partial f_n(s_k)}{\partial \{s\}_n} \right) \\ l_{i,sk}(s) = \nabla f_i(s_k) \cdot (s - s_k) + f_i(s_k) = \sum_{j=1}^{n} \frac{\partial f_i(s_k)}{\partial \{s\}_j} \cdot \{(s - s_k)\}_j + f_i(s_k) \end{gathered} \tag{9.11}$$

Dado que un polinomio de Taylor aproxima una función en el entorno de un punto, es de esperar que, si s_k es *suficientemente cercano* a las variedades en $\Re^n$ dadas por los cortes con cero de cada $f_i(s)$, dichas variedades estén aproximadas por las variedades afines dadas por los cortes de cada aproximación por Taylor (de grado 1) de $f_i(s)$ en s_k. Por lo tanto, cada iteración vendrá dada por la resolución de $l_{i,sk}(s) = 0$ $\forall i \in [1, n]$, Es decir, $l_{i,sk}(s)$. Si se crea un sistema con las distintas aproximaciones a $f_i(s)$ en s_k, $l_{i,sk}(s)$, igualadas a cero, se tiene que:

$$\begin{cases} \nabla f_1(s_k) \cdot (s - s_k) + f_1(s_k) = 0 \\ \nabla f_2(s_k) \cdot (s - s_k) + f_2(s_k) = 0 \\ \vdots \\ \nabla f_n(s_k) \cdot (s - s_k) + f_n(s_k) = 0 \end{cases} \tag{9.12}$$

Esto se puede escribir en una expresión matricial única empleando *F* y el jacobiano de *F* en s_k, $J_F(s_k)$, cuyas filas son los gradientes en s_k de cada función coordenada f_i.

$$J_F(s_k) \cdot (s - s_k) + F(s_k) = 0 \tag{9.13}$$

Como antes, la resolución de s dará el valor del punto en la siguiente iteración: s_k.

$$s_{k+1} = s_k - \left(J_F(s_k)\right)^{-1} \cdot F(x_k) \tag{9.13}$$

Que es la resolución de un sistema lineal *n x n* seguido de la resta de dos vectores de *n* elementos.

Un punto fuerte del método de Newton-Raphson es que su convergencia es en general de orden dos. Podría ser menor si la multiplicidad de la raíz fuese mayor que uno, pero esto puede tratarse como excepción puntual en un polinomio con coeficientes arbitrarios. Sin embargo, su comportamiento, como el del punto fijo, depende enormemente del punto inicial de búsqueda, ya que se basa, como ya se ha mencionado, en la capacidad que tiene un polinomio de Taylor de grado 1 de aproximar una función no lineal, y en general la aproximación por polinomios de Taylor es efectiva en un entorno local del punto donde se evalúa la función y su derivada (o sus derivadas, si es de grado mayor que 1). También depende de la naturaleza de la función aproximada, ya que para condiciones iguales, algunas funciones son más "previsibles" en función de sus derivadas que otras. En general, si el método no converge a la solución buscada, puede que lo haga a otra solución del sistema que no sea la deseada, o directamente diverja, yendo sus valores al infinito. Por lo observado, esto sucede frecuentemente cuando el gradiente de algunas de las funciones coordenadas en un punto de la iteración es cercano a cero, lo que provoca que la siguiente iteración se aleje bruscamente, entrando en un régimen impredecible. Esto se puede emplear para introducir una condición de parada que evite las subsiguientes iteraciones, en general inútiles. Aunque en teoría un bucle finito de valores sería posible, la realimentación positiva de las diferencias entre dos puntos de iteración muy similares hace de la localización de un caso de este fenómeno un problema complejo por sí mismo.

Método del descenso más rápido.

Este es el tercero considerado para la implementación. Se basa en el siguiente razonamiento: Si resolver el sistema $F(s)=0$ consiste en hallar el punto $\hat{s}\in\Re^n$ tal que $f_i(\hat{s})=0 \ \ \forall i\in[1,n]$, entonces $\hat{s}$ será también la única solución del sistema:

$$S(s)=0 \qquad S:\Re^n\to\Re \ / \ S(s)=\sum_{i=1}^{n} f_i(s)^2 \tag{9.14}$$

Además, el valor del sumatorio en este punto será el mínimo absoluto de la función *S(s)*. Por tanto, el problema se puede traducir a la búsqueda de dicho mínimo:

$$\hat{s}=\arg\min_{s} S(s) \tag{9.15}$$

Empleando el hecho de que, para un punto $s\in\Re^n$ la derivada direccional de $S(s)$ presenta el máximo cuando ésta apunta en la dirección del gradiente, se deduce que el máximo decrecimiento para el valor de la función en un punto se obtendrá en la dirección opuesta al gradiente. Por tanto, un método iterativo que en cada iteración siga esa dirección, tendrá visos de llegar a un mínimo de algún tipo. Existen muchas variantes, algunas simulando el movimiento inercial de una partícula masiva que se desplazara sobre la superficie en $\Re^{n+1}$ dada por $S(s)$. El método del descenso más rápido intenta hacer más trabajo en cada iteración, buscando un mínimo en la función:

$$h:\Re\to\Re \qquad h(\alpha)=s_k-\alpha\cdot\nabla S(s_k) \tag{9.16}$$

Para hallarlo, en cada iteración debe construirse la función *h*, y se debe emplear el procedimiento típico de búsqueda de raíces en la derivada, más las posteriores comprobaciones. Esto puede ser bastante costoso para un proceso iterativo, por lo que un procedimiento habitual es calcular el valor de la función en tres puntos: $\alpha_1, \alpha_2, \alpha_3$ elegidos donde se espera que la función se minimice, crear el polinomio cuadrático que interpola dichos puntos, y hallar el mínimo de dicho polinomio, que es una tarea simple dado que la derivada es lineal. $\alpha_1, \alpha_2, \alpha_3$ se pueden elegir tomando $\alpha_1 = s_k$, de forma que ya esté calculado, y tomando α_2, α_3 equidistantes y en la dirección contraria a $\nabla S(s_k)$, como se ha comentado.

Existen multitud de variantes para este método, especialmente en lo que concierne a la minimización de $h(\alpha)$, pero en general, según Burden y Douglas, todas tienen grado de convergencia 1. Por otro lado, si bien el método no diverge al infinito como los dos anteriores, sí puede quedarse estancado en mínimos locales formados por los mínimos absolutos de cada $f_i(s)^2$. Esto parece aún más probable si se emplea la técnica de interpolación polinómica para $h(\alpha)$. Los mínimos locales son puntos de convergencia que no son soluciones del sistema *n x n*, y es un problema específico de este método. Por otro lado, si *F(s)=0* tiene un número finito de soluciones, $S(s)$ tendrá varios mínimos absolutos iguales a cero, y el sistema también podrá converger a cualquiera de ellas, como ya sucedía en los anteriores métodos.

Comparación de métodos

A continuación se discutirá la elección de un método para la resolución del sistema (9.5).

El método del punto fijo queda descartado por no ofrecer una metodología concreta para la obtención de la función $G(s)$, lo cual puede ser causa de divergencia por no cumplirse las condiciones estipuladas para sus derivadas parciales. Por este motivo varios autores lo desaconsejan. Además, aunque en la literatura sólo se ha encontrado referencia explícita a que el grado de convergencia del caso de resolución de ecuaciones es 1, es razonable suponer que en el caso de resolución de sistemas también lo será, lo cual lo convertiría en un método lento.

El método de Newton, por su parte, puede presentar ciertos problemas de complejidad computacional. Cuando las derivadas parciales son expresiones mucho más complejas que las funciones coordenadas originales, lo cual sucede frecuentemente cuando se aplica la regla de la cadena un elevado número de veces, suelen emplearse métodos alternativos que evitan el cálculo directo de las derivadas. En el caso de la resolución de ecuaciones, se puede emplear el de la secante, que determina *l(x)* mediante las imágenes de las dos últimas iteraciones. En el caso de la resolución de sistemas, el efecto se ve agravado por el hecho de que son n^2 derivadas parciales. Para este caso, en algunas ocasiones en las que no es posible el cálculo previo de las derivadas mediante motores de cálculo simbólico, se pueden emplear diferencias finitas. Sin embargo, esto no evita que la inversión de la matriz del sistema tenga complejidad asintótica $O(n^3)$, por lo que para sistemas de grandes dimensiones, hay que acudir a métodos conocidos como cuasi-Newton, que reemplazan la matriz jacobiana por una aproximación que se actualiza en

cada iteración. Un ejemplo de estos es el de Broyden, que generaliza el método de la secante y consigue una complejidad asintótica total $O(n)$, pero cuyo grado de convergencia se reduce a 1. Otro problema de los métodos cuasi-Newton es que no corrigen los errores de redondeo en las iteraciones sucesivas, salvo que se incorporen medidas especiales (Burden y Douglas, 2002).

Sin embargo, en el caso del sistema (9.5), las ecuaciones y sus correspondientes funciones son polinómicas de grado 4, por lo que las derivadas parciales son en general expresiones más sencillas de evaluar (y su obtención también es sencilla, aunque lo contrario tampoco sería un problema disponiendo un sistema de cálculo simbólico). Por otro lado, la dimensión del sistema es *5x5,* que no requiere de técnicas especiales y el sistema lineal en cada iteración se puede resolver directamente. Por lo tanto, puede emplearse el método de Newton-Raphson básico sin recurrir a variantes que limiten su precisión arbitraria (hasta los límites del sistema de representación numérica empleado) o su velocidad de convergencia de grado 2, lo cual supone un aliciente para elegir este método.

Por último, el método del descenso más rápido ofrece la ventaja de que no diverge a infinito, pero la desventaja de que puede hacerlo a mínimos locales que surgen de mínimos absolutos de subconjuntos del conjunto total de funciones coordenadas sumadas cuadráticamente. Además, su convergencia es de grado 1. En cuanto a la complejidad inherente a cada iteración, debe calcular un gradiente de *n* términos en lugar de un jacobiano con n^2, pero cada derivada parcial es una expresión más compleja debido a la suma cuadrática (aunque sigue siendo derivación polinómica). En cualquier caso, esta complejidad depende enormemente del método seguido tras la evaluación del gradiente, a la hora de minimizar $h(\alpha)$.

Hay otro punto a tener en cuenta en la comparación entre el método de Newton-Raphson (NR) y el de descenso más rápido (DMR), que es particular al caso que se está tratando. Como ya se ha comentado con anterioridad, aunque en algunos momentos se ha omitido mencionar el hecho, el sistema tal como está planteado en (9.5) presenta múltiples soluciones, y cualquiera de los métodos presentados podrá converger a cualquiera de ellas, dependiendo de la situación relativa de s_0. La forma obvia de establecer si la solución es la correcta es probar si satisface la ecuación extra, una vez el método ha convergido. Esto obliga a repetir el método con un conjunto de puntos iniciales repartidos en el espacio de búsqueda, de forma que se garantice con suficiente frecuencia la convergencia a la solución válida con alguno de ellos. Esto tampoco podría evitarse aunque no existiera este fenómeno, ya que ambos métodos pueden acabar no convergiendo a la solución del sistema aunque esta sea única (NR puede diverger al infinito, DMR puede hacerlo a mínimos locales), pero es una fuente añadida de falsas soluciones, y es de esperar que tenga una repercusión importante sobre el tiempo medio tomado por el sistema para hallar la solución correcta.

La situación de las falsas soluciones es demasiado arbitraria, dependiendo del caso concreto, de modo que no se ha encontrado ningún método sencillo de establecer, a través de la información completa del problema en un caso concreto, un punto inicial suficientemente cercano a la solución válida. Hay que señalar que los teoremas relacionados con los métodos iterativos suelen ofrecer únicamente seguridad sobre la existencia de "un entorno" dentro del cual se converge a una solución, y esta es única.

Los posibles métodos para cuantificar este entorno para un caso concreto pertenecen a otro nivel de complejidad, y no se hace referencia a ellos en la literatura consultada.

El único remedio pasaría entonces por poder modificar el método iterativo de forma que incluya la información dada por la sexta ecuación de forma semejante a como está incluida la de las otras cinco, de modo que el método iterativo converja a la única solución que tienen las seis ecuaciones en común. En otras palabras, emplear un método iterativo sobredeterminado.

Para el caso de NR, se pueden idear muchas variantes que intuitivamente van en esta dirección. Un ejemplo sería calcular la aproximación lineal de las seis funciones componentes y resolver el sistema lineal sobredeterminado resultante. Por supuesto, el hecho de que seis funciones $\Re^5 \rightarrow \Re$ ofrezcan una solución común no significa que sus aproximaciones lineales lo hagan. Aun así, podría encontrarse una solución de mínimo error cuadrático calculando la pseudoinversa de Moore-Penrose del Jacobiano *6x5* resultante.[3] Sin embargo, no se ha encontrado ninguna prueba matemática sólida de que el resultado de este proceso se acerque necesariamente a la solución común de las seis funciones componentes. Podría darse el hecho de que un punto s_k determinado estuviera en la zona de convergencia de una solución para un subconjunto de 5 funciones coordenadas de entre las 6 totales, pero para la de otra solución diferente para otro subconjunto distinto. El resultado de esto puede ser imprevisible a este nivel de análisis teórico, de modo que sólo podría obtenerse una conclusión mediante resultados experimentales.

En el caso de DMR, la incorporación de la sexta ecuación al sumatorio cuadrático sí produce la reducción de los mínimos absolutos de *S(s)* a uno, también de valor cero, de modo que parece una opción más segura. Aún así, se aumenta el número de mínimos locales, al aumentar el número de subconjuntos propios de funciones coordenadas.

Con el análisis hecho hasta ahora, se puede elaborar una tabla comparativa de los métodos NR y DMR, simplificando sus características más generales:

	Divergencia	Convergencia a soluciones ajenas a F(s)=0	Modificación para solución única	Grado de convergencia	Complejidad de cada iteración
NR	Sí	No	Posible	2	Evaluación de 25 polinomios de grado 4, resolución de sistema lineal 5x5
DMR	No	Sí	Sí	1	Evaluación de 6 polinomios de grado 8, resto dependiente de implementación

Tabla 9.1

La tabla es meramente orientativa, ya que hay muchas variables en juego. La decisión tomada ha sido implementar la versión tradicional de Newton-Raphson, y en función de los resultados obtenidos, probar o no la implementación del descenso más rápido.

[3] Para una descripción formal de la pseudoinversa de Moore-Penrose y su relación con la solución mínimocuadrática de sistemas lineales sobredeterminados, puede consultarse Merino y Santos (1997).

Estos resultados han sido que, aunque los casos de divergencia y convergencia a soluciones no válidas son frecuentes, empleando una batería de puntos iniciales y unas condiciones de parada adecuadas, la resolución es correcta para una gran proporción de casos, y el tiempo tomado en un procesador comercial estándar es de menos de 10 segundos en el peor caso, con una media cercana al segundo, algo aceptable teniendo en cuenta que es una tarea de calibrado y no de localización en tiempo real.

Hay que resaltar que debido a las múltiples fuentes de imprecisión en la posición de las primitivas bidimensionales, el sistema sobredeterminado no lo será en un sentido matemáticamente ideal. Por ello, al probar distintas soluciones con la sexta ecuación, el resultado no será en general coherente ni siquiera cuando se trate de la solución válida. Por ello es más conveniente emplear el operador mínimo para establecer dicha solución. Por otro lado, al considerarse el resultado de dicho operador como una medida de coherencia de la solución, con él se puede tomar una decisión, máquina o humana (en la implementación realizada es humana) sobre la conveniencia de repetir el proceso de calibrado.

En la figura (9.4) se incluye el pseudocódigo del algoritmo empleado en la implementación para repetir el método de Newton-Raphson sobre una batería de puntos iniciales de búsqueda, y elegir la mejor solución. Para una mayor claridad, se declaran independientemente las variables de entrada y las que que intervienen en el funcionamiento y parada tanto de este algoritmo como del de Newton-Raphson.

La batería de puntos empleada es holgada, en el sentido de que las combinaciones completas que hace sobre los 4 parámetros permiten que se recorra homogéneamente todo el espacio de posibles soluciones (con una densidad marcada por la magnitud de los saltos). Aunque podría emplearse un método más elaborado, que tuviera en cuenta la geometría del patrón de calibrado para evitar combinaciones de parámetros absurdas, este algoritmo es sencillo de implementar, ligero (en lo que a la elaboración de la batería respecta), y con las constantes descritas se ha demostrado suficientemente veloz sobre una máquina estándar. Tampoco ha demostrado mayor tasa de fracasos que en versiones más exhaustivas (mayor rango de búsqueda, valores de salto menores, condiciones de parada más estrictas).

```
 1 //DATOS INICIALES
 2 distanciasPatron = <distancias mutuas entre puntos 3d patron>
 3 minDistCamaraPatron = <distancia minima entre camara y patron>
 4 maxDistCamaraPatron = <distancia maxima entre camara y patron>
 5 puntosPlano = <coordenadas de los puntos del patron proyectados>
 6 virtualH = <altura en pixeles de la imagen>
 7 virtualW = <anchora en pixeles de la imagen>
 8
 9 //VARIABLES QUE CONTROLAN BUCLE SOBRE
10 //BATERIA DE PUNTOS INICIALES
11 diagVirtual = sqrt(virtualH^2+ virtualW^2)
12 minBusqLM = minDistCamaraPatron/diagVirtual
13 maxBusqLM = maxDistCamaraPatron/diagVirtual
14 stepLM = (maxBusqLM-minBusqLM)/4
15 minBusqDF = diagVirtual*0.8*0.5
16 maxBusqDF = diagVirtual*0.8*1.5
17 stepDF = (maxBusqDF-minBusqDF)/4
18 maxIncoherenciaAceptable = 2;
19
20 //VARIABLES DE CONDICION DE PARADA DE NEWTON-RAPHSON
21 errorMaxPermitido = 0.01
22 numMaxIteraciones = 20
23
24 //BUCLE SOBRE BATERIA DE PUNTOS INICIALES
25 incoherenciaMin = +Inf
26 FOR lambda1 FROM minBusqLM TO maxBusqLM STEPPING stepLM
27   FOR lambda2 FROM minBusqLM TO maxBusqLM STEPPING stepLM
28     FOR lambda3 FROM minBusqLM TO maxBusqLM STEPPING stepLM
29       FOR lambda4 FROM minBusqLM TO maxBusqLM STEPPING stepLM
30         FOR distFocal FROM minBusqDF TO maxBusqDF STEPPING stepDF
31           BEGIN
32           vector_inicial =
33              [lambda1 lambda2 lambda3 lambda4 distFocal]
34           [vectorFinal error] = NewtonRaphson(
35              ,numMaxIteraciones,errorMaxPermitido,
36              ,vectorInicial,puntosPlano,distanciasPatron)
37            incoherencia = calculaIncoherencia(
38               ,vectorFinal, puntosPlano)
39           IF error<errorMaxAdmitido AND incoherencia<incoherenciaMin
40            BEGIN
41            incoherenciaMin = incoherencia
42            vectorSolucion = vectorFinal
43            IF incoherencia<maxIncoherenciaAceptable
44              BREAK ALL LOOPS
45            END
46           END
```

Figura 9.4

La explicación de las líneas 12 y 13 está en que, como se comentó en el apartado "Modelo pinhole de una cámara", tanto en este texto como en el código de la implementación se considera un modelo de cámara donde el plano imagen tiene dimensiones físicas dadas por la relación 1 píxel = 1 mm^2. De la misma forma, la distancia focal, salvo que se indique lo contrario, se maneja en relación a ese tamaño de plano imagen. Esta convención libera de ambigüedades el manejo de la distancia focal, y permite una escritura más sencilla y eficiente de las ecuaciones (9.5) en el código interno de la función `NewtonRaphson()`. Las líneas 12 y 13, por tanto, construyen los límites de los coeficientes lambda de las ecuaciones (9.5) en función de los límites

de distancia del objetivo de la cámara al patrón (que se espera que el usuario respete, ver Apéndice), y en función de las dimensiones del plano imagen.

Las líneas 15 y 16 establecen un intervalo de búsqueda aproximadamente centrado en una distancia focal típica, correspondiente a un objetivo normal, que equivale a cerca de 0.8 veces la diagonal del sensor.

El resto de constantes que aparecen en el pseudocódigo han sido seleccionadas manualmente para obtener un rendimiento aceptable en el proceso de búsqueda global.

En lo que respecta a Newton-Raphson, se ha empleado un valor para `errorMaxPermitido` relativamente alto, ya que considerando que la solución del sistema 5x5 va a poseer cierto grado de incoherencia con la sexta ecuación, es inútil perseguir una precisión muy alta en la solución, y evitándolo se puede agilizar el proceso. El valor elegido para `numMaxIteraciones` es el más bajo que ha demostrado superar el número de iteraciones que se toman la mayoría de ejecuciones convergentes, para el valor de `errorMaxPermitido` dado. No se incluye pseudocódigo para el método de Newton-Raphson y las funciones desde él llamadas, porque su descripción está incluida en el análisis previo: la ecuación (9.13) describe el núcleo del método de Newton-Raphson, (9.6.bis) describe la condición de parada, de las ecuaciones (9.5) se extrae fácilmente $F(x)$, y su jacobiano $J_F(x)$ está descrito de forma general en las ecuaciones (9.11) (9,12) (9.13). Su expresión exacta puede obtenerse mediante el proceso rutinario de derivación de las ecuaciones polinómicas de $F(x)$ en (9.5), resultando en unas nuevas ecuaciones polinómicas ligeramente más extensas. Por supuesto, en la implementación, estas expresiones polinómicas de $J_F(x)$ se han introducido como *hardcode* de la misma forma que las de $F(x)$, evitando la sobrecarga inútil que supondría la derivación mediante cálculo simbólico en cada iteración interna de Newton-Raphson.

En lo que respecta al bucle que controla la batería de puntos iniciales, las constantes en el denominador de las líneas 14 y 17 han demostrado un buen rendimiento. Como se ha dicho anteriormente, no se ha observado un aumento significativo de la tasa de fracasos aumentando la densidad de los puntos de búsqueda por encima de lo establecido por esas constantes. Y de esta forma, el tiempo tomado por el proceso global de búsqueda estará acotado. Para estas constantes, con una máquina actual estándar, como ya se mencionó, está alrededor de los 10 segundos. Sin embargo, la condición de la línea 43, marcada por la variable manual de la línea 18, suele saltar rápidamente en la mayoría de los casos en el que el resultado final es aceptable, de modo que en media, el tiempo tomado por el proceso general de búsqueda es menor, y mucho menor (del orden de 1 segundo) si se consideran únicamente los resultados con una coherencia aceptable.

La función `calculaIncoherencia()` , por otro lado, emplea el miembro izquierdo de la sexta ecuación de (9.5), redundante en el sistema 6x5.

Condicionamiento del sistema

Debido a las dimensiones del patrón de calibrado empleado, se ha observado que cuando se sitúa a la distancia necesaria para que se forme un espacio amplio visible para todas las cámaras calibradas, el sistema (9.5) queda muy mal condicionado. Concretamente, para desviaciones mínimas en las coordenadas discretas de las primitivas bidimensionales, se produce una gran variación del valor estimado de la distancia focal f y otra de magnitud aproximadamente inversa en el valor de las λ_i. Expresado en términos fotográficos: los cambios en la imagen del patrón cuando se aleja la cámara y se aumenta el zoom son muy pequeños a partir de cierta distancia entre cámara y patrón, que depende de las dimensiones del último. Esto provoca que el error introducido por las distorsiones de la cámara, la discretización de la imagen y el propio método para localizar las primitivas bidimensionales se amplifique en la estimación final de la distancia focal y las primitivas tridimensionales. Dado que aumentar las dimensiones del patrón no es práctico, se ha optado por implementar esta fase en dos versiones:

Una, tal como se ha explicado hasta ahora, empleando 4 puntos, (5+1) ecuaciones, y 5 incógnitas: $(\lambda_1, \lambda_2, \lambda_3, \lambda_4, f)$. Al acercar el patrón a una cámara, de forma que ocupe la mayor parte de su campo visual, el mal condicionamiento antes descrito se reduce, por lo que pueden evaluarse las cinco incógnitas con mayor precisión. Sin embargo, no se puede acercar el patrón a todas las cámaras simultáneamente sin mover estas y modificar el área de enfoque común, por lo que este procedimiento ha de hacerse individualmente para cada cámara, y el único parámetro aprovechable es el que no varía por ello: el instrínseco f de cada cámara.

Una vez se dispone de la distancia focal de cada cámara calculada mediante un sistema mejor condicionado, se puede realizar el proceso de calibrado simultáneo para todas las cámaras. Si se retorna a la discusión sobre la ecuación (9.3), se puede ver que empleando tres puntos ahora hay tres ecuaciones con términos independientes $(d_{1,2}, d_{1,3}, d_{2,3})$ y tres incógnitas $(\lambda_1, \lambda_2, \lambda_3)$, por lo que el sistema ya es resoluble, y ya no existe el problema de condicionamiento anterior porque f está fijada. Para este sistema se puede emplear el mismo método de Newton-Raphson, que además requerirá por lo general menos iteraciones, por lidiar con un jacobiano *3x3* , y tener que probar un número mucho menor de puntos iniciales de búsqueda. Aunque en esta ocasión no se resuelve p_4, sí que se usa para establecer la medida de coherencia que determinará cuál de las múltiples soluciones a las que converge ahora NR es la correcta. Antes se describió este sistema como la intersección de tres cilindros con características concretas en el espacio tridimensional, de forma que puede visualizarse mejor la existencia, aquí también, de múltiples soluciones.

Una forma de establecer la medida de coherencia es determinar la posición de p_4 en función de la de p_1, p_2, p_3 y las distancias conocidas a priori $d_{1,4}, d_{2,4}, d_{3,4}$, y posteriormente calcular la distancia entre p_4 y la recta $\overline{op_{p4}}$, que es la recta de todos los puntos que podrían producir la proyección p_{p4}. Sin embargo, el cálculo de p_4 requiere de esta forma la resolución de otro sistema no lineal (intersección de tres esferas). En cambio, si se emplea un patrón que define p_1, p_2, p_3, p_4 como coplanarios,

se puede construir una medida de coherencia igualmente válida, pero mediante métodos lineales directos mucho más ligeros computacionalmente. Se define p'_4 como la intersección del plano dado por p_1, p_2, p_3 y la recta $\overline{op_{p4}}$, que viene a ser una estimación de p_4 donde prima la información de coplanariedad y de $\overline{op_{p4}}$ sobre la de las distancias mutuas entre puntos. Entonces la medida se puede establecer como:

$$c = \sum_{i=1}^{3} \left| d(p_i, p'_4) - d(p_i, p_4) \right| \tag{9.17}$$

Esta medida de coherencia, igual que en el caso *5x5*, en general no será cero ni siquiera para la solución correcta, y se puede emplear tanto para discernir la solución convergente correcta de NR, como para determinar la calidad de ésta.

El hecho de que p_1, p_2, p_3, p_4 sean coplanarios no supondrá un problema a la hora de calcular el movimiento rígido entre cámaras, como se verá en el siguiente apartado.

Fase 3 de calibrado: Cálculo de la posición relativa entre cámaras

Una vez se tienen las coordenadas de los puntos del patrón en función de los distintos sistemas de referencia asociados a cada una de las cámaras, se ha de calcular la matriz de paso entre dichos sistemas de referencia. Como ya se comentó antes, se ahorran cálculos si se selecciona una de las cámaras existentes para que su sistema de referencia asociado sea el que se emplee como absoluto en la escena (con el que se describirá la posición de v), en vez de elegir uno externo asociado a una cámara imaginaria C_0, como se hizo inicialmente. Por convención, la cámara cuyo sistema de referencia asociado sea el absoluto se denotará como C_1. Por tanto, la labor de este módulo, para n cámaras, puede representarse de la siguiente forma:

$$\begin{Bmatrix} (p_1)_{R1} & (p_2)_{R1} & (p_3)_{R1} & (p_4)_{R1} \\ (p_1)_{R2} & (p_2)_{R2} & (p_3)_{R2} & (p_4)_{R2} \\ \vdots & & & \\ (p_1)_{Rn} & (p_2)_{Rn} & (p_3)_{R1} & (p_4)_{Rn} \end{Bmatrix} \rightarrow \begin{Bmatrix} M_1 \\ M_2 \\ \vdots \\ M_n \end{Bmatrix} \tag{10.1}$$

Cumpliendo:

$$M_k \cdot (p_j)_{Rk} = (p_j)_{R1} \qquad \forall j \in [1,4] \tag{10.2}$$

Donde $(p_j)_{Rk}$ es el punto i expresado en el sistema de referencia de la cámara k, y M_k es la matriz de conversión del sistema de referencia asociado a la cámara k al absoluto. Como consecuencia de emplear como sistema de referencia global el asociado a C_1, se tiene que:

$$M_1 = \left[\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \tag{10.3}$$

Es decir, la transformación identidad en el espacio afín de tres dimensiones. Por tanto, sólo hay que hallar $n-1$ matrices.

Asumiendo de momento que p_1, p_2, p_3, p_4 son no coplanarios, se han elaborado dos métodos para, considerando la ecuación (10.2), definir M_k.

El primero consiste en reescribir las 12 ecuaciones lineales que resultan de aplicar la ecuación (10.2) para los 4 puntos en una sola expresión matricial. Si se emplea la siguiente notación:

$$M_k = \left[\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline t_1 & r_{1,1} & r_{1,2} & r_{1,3} \\ t_2 & r_{2,1} & r_{2,2} & r_{2,3} \\ t_3 & r_{3,1} & r_{2,3} & r_{3,3} \end{array}\right] \tag{10.4}$$

Cada una de las 12 ecuaciones se puede escribir de forma genérica de la siguiente forma:

$$t_i + \{(p_j)_{Rk}\}_1 \cdot r_{i,1} + \{(p_j)_{Rk}\}_2 \cdot r_{i,2} + \{(p_j)_{Rk}\}_3 \cdot r_{i,3} = \{(p_j)_{R1}\}_i \quad \forall i = 1,2,3 \quad \forall j = 1,2,3,4 \tag{10.5}$$

Se puede observar que es un conjunto de ecuaciones lineales, donde los elementos no triviales de M_k se pueden considerar las incógnitas. En concreto, se pueden organizar en el siguiente vector de $\Re^{12}$:

$$\breve{m}_k = \begin{bmatrix} t_1 & r_{1,1} & r_{1,2} & r_{1,3} \mid t_2 & r_{2,1} & r_{2,2} & r_{2,3} \mid t_3 & r_{3,1} & r_{3,2} & r_{3,3} \end{bmatrix}^t \tag{10.6}$$

Con la información proporcionada por los puntos se puede construir una matriz 12x12 y un vector de términos independientes que desemboquen en un único sistema de ecuaciones lineales. Además, si se ordenan las incógnitas por filas de M_k, como se ha hecho en la construcción de $\breve{m}_k$, se puede ver que los coeficientes que los multiplican son los mismos, lo que resulta en que la matriz del sistema resultante, denominada $\breve{P}_k$, quede en forma diagonal por bloques 4x4:

$$\breve{P}_k = \begin{bmatrix} B & & \\ & B & \\ & & B \end{bmatrix} \in \mathrm{M}_{12\times 12}(\Re) \qquad B = \begin{bmatrix} 1 & \{(p_1)_{Rk}\}_1 & \{(p_1)_{Rk}\}_2 & \{(p_1)_{Rk}\}_3 \\ 1 & \{(p_2)_{Rk}\}_1 & \{(p_2)_{Rk}\}_2 & \{(p_2)_{Rk}\}_3 \\ 1 & \{(p_3)_{Rk}\}_1 & \{(p_3)_{Rk}\}_2 & \{(p_3)_{Rk}\}_3 \\ 1 & \{(p_4)_{Rk}\}_1 & \{(p_4)_{Rk}\}_2 & \{(p_4)_{Rk}\}_3 \end{bmatrix} \tag{10.7}$$

De forma análoga, se construye el vector de términos independientes con las coordenadas de los 4 puntos en el sistema de referencia R1:

$$\breve{p}_k = \begin{bmatrix} \{(p_1)_{R1}\}_1 & \{(p_2)_{R1}\}_1 & \{(p_3)_{R1}\}_1 & \{(p_4)_{R1}\}_1 & \cdots \\ \cdots & \{(p_1)_{R1}\}_3 & \{(p_2)_{R1}\}_3 & \{(p_3)_{R1}\}_3 & \{(p_4)_{R1}\}_3 \end{bmatrix}^t \tag{10.8}$$

El sistema resultante se puede expresar en la forma:

$$\breve{M}_k \cdot \breve{m}_k = \breve{p}_k \tag{10.9}$$

Y se puede aplicar cualquier método tradicional de resolución de sistemas lineales, como Gauss-Jordan. Al ser un sistema 12x12, el número de operaciones no es despreciable, pero tampoco supone un problema si no hay requisitos temporales muy estrictos, como sería el caso en tiempo real.

El segundo método ideado consiste en emplear p_1, p_2, p_3, p_4 para la construcción de un sistema de referencia intermedio, $RP = \{p_1; \overrightarrow{p_1p_2}, \overrightarrow{p_1p_3}, \overrightarrow{p_1p_4}\}$, y con las coordenadas de

p_1, p_2, p_3, p_4 en función de *R1* y *RK*, construir directamente las matrices de paso de *RP* a *R1* y *RP* a *RK* respectivamente:

$$M_{RP\to R1} = \left[\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline \{(p_1)_{R1}\}_1 & \{\overrightarrow{(p_1)_{R1}(p_2)_{R1}}\}_1 & \{\overrightarrow{(p_1)_{R1}(p_3)_{R1}}\}_1 & \{\overrightarrow{(p_1)_{R1}(p_4)_{R1}}\}_1 \\ \{(p_1)_{R1}\}_2 & \{\overrightarrow{(p_1)_{R1}(p_2)_{R1}}\}_2 & \{\overrightarrow{(p_1)_{R1}(p_3)_{R1}}\}_2 & \{\overrightarrow{(p_1)_{R1}(p_4)_{R1}}\}_2 \\ \{(p_1)_{R1}\}_2 & \{\overrightarrow{(p_1)_{R1}(p_2)_{R1}}\}_3 & \{\overrightarrow{(p_1)_{R1}(p_3)_{R1}}\}_3 & \{\overrightarrow{(p_1)_{R1}(p_4)_{R1}}\}_3 \end{array}\right]$$

$$M_{RP\to RK} = \left[\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline \{(p_1)_{RK}\}_1 & \{\overrightarrow{(p_1)_{RK}(p_2)_{RK}}\}_1 & \{\overrightarrow{(p_1)_{RK}(p_3)_{RK}}\}_1 & \{\overrightarrow{(p_1)_{RK}(p_4)_{RK}}\}_1 \\ \{(p_1)_{RK}\}_2 & \{\overrightarrow{(p_1)_{RK}(p_2)_{RK}}\}_2 & \{\overrightarrow{(p_1)_{RK}(p_3)_{RK}}\}_2 & \{\overrightarrow{(p_1)_{RK}(p_4)_{RK}}\}_2 \\ \{(p_1)_{RK}\}_2 & \{\overrightarrow{(p_1)_{RK}(p_2)_{RK}}\}_3 & \{\overrightarrow{(p_1)_{RK}(p_3)_{RK}}\}_3 & \{\overrightarrow{(p_1)_{RK}(p_4)_{RK}}\}_3 \end{array}\right] \tag{10.10}$$

Así, la matriz de paso de RK a R1 se puede obtener de la siguiente forma:

$$M_k = \left(M_{RP\longrightarrow RK}\right)^{-1} \cdot M_{RP\longrightarrow R1} \tag{10.11}$$

Conviene notar que *RP* no tiene por qué ser rectangular (con base asociada ortonormal). En el caso general no lo será, por lo que las matrices de los cambios de base asociados a los cambios de referencia intermedios no serán necesariamente ortonormales. Sin embargo, el producto descrito, por ser *RK* y *R1* rectangulares, ha de contener un cambio de base ortonormal.

Con los dos métodos descritos, el problema parece resuelto. Sin embargo, hay una peculiaridad que no se ha mencionado: por el patrón elegido en la implementación de la fase 1 de calibrado (que se explicará a continuación de ésta), p_1, p_2, p_3, p_4 son coplanarios, y por tanto afínmente dependientes. Esto provoca que los vectores de la base asociada sean linealmente dependientes y la matriz de paso singular, por lo que no se puede realizar la inversión en la ecuación (10.11). La razón de que no se haya comentado antes es que es algo impuesto por una elección concerniente al siguiente módulo, y ajena al caso general de este. En cualquier caso el problema tiene solución, ya que de lo contrario se habría cambiado la elección tomada en la fase 1.

La solución empleando el primer método descrito pasaría por eliminar un punto, que es afínmente dependiente de los otros 3, y rebajar los grados de libertad de la definición de M_k en la ecuación (10.4), empleando la información conocida de que la matriz de cambio de base asociada es ortonormal, por serlo las bases asociadas a RK y R1. Sin embargo, esto obliga a introducir funciones trigonométricas en los coeficientes de M_k, y el sistema se vuelve irresoluble linealmente.

La solución por el segundo método supone una complicación adicional mucho menor, y unido a su sencillez inicial, es lo que ha hecho que se utilice en la implementación del

sistema. La labor de esta fase puede interpretarse como determinar un movimiento rígido que ajuste dos versiones idénticas de un poliedro de cuatro vértices en posiciones y orientaciones distintas (que es el mismo poliedro, desde sistemas de referencia rectangulares distintos). Lo que se tiene en el caso coplanario es dos versiones de un polígono de tres vértices, cada una con distinta posición y orientación en el espacio, y se necesita añadir un cuarto punto a cada una para formar dos versiones de un mismo poliedro. En otras palabras, para cada triángulo, hay que construir un cuarto punto que no sea coplanario, y que en cada caso esté en la misma posición relativa con respecto a los otros tres. Que esté contenido en la recta perpendicular al plano generado por $\overrightarrow{p_1p_2}$ y $\overrightarrow{p_1p_3}$ y que pasa por p_1, y que esté a la misma distancia de p_1 es condición suficiente para esto. Esto se puede conseguir con un producto vectorial apoyado sobre p_1: el punto $p_4' = p_1 + \overrightarrow{p_1p_2} \times \overrightarrow{p_1p_3}$ es el que se está buscando.

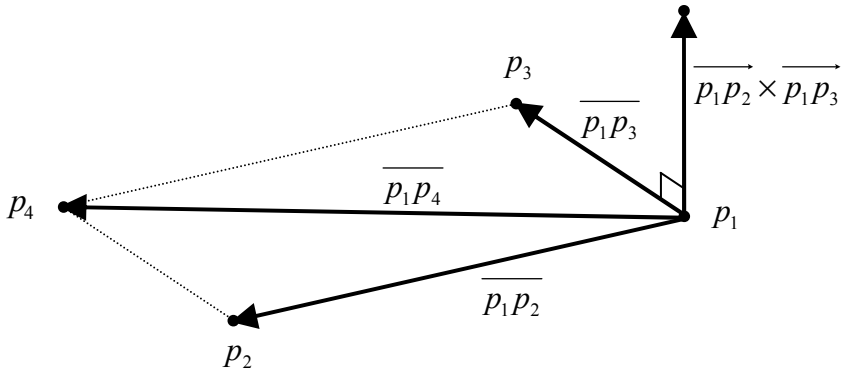


Figura 10.1

Además, como puede apreciarse en la figura (10.1), resulta que los puntos se pueden elegir de forma que la base $(\overrightarrow{p_1p_2}, \overrightarrow{p_1p_3}, \overrightarrow{p_1p_2} \times \overrightarrow{p_1p_3})$ sea ortogonal (no ortonormal), porque $\overrightarrow{p_1p_2} \cdot \overrightarrow{p_1p_3} = 0$. Pero esto es nuevamente una consecuencia de la elección del patrón en la fase 1 (los vértices de un rectángulo), y el método sería igualmente válido si $\overrightarrow{p_1p_2} \cdot \overrightarrow{p_1p_3} \neq 0$.

Como último comentario respecto al diseño de esta fase, conviene recordar que los datos recibidos no serán exactos, ya que se arrastrará un error de la fase 2, que provendrá mayoritariamente de un error en la fase 1, debido a distorsiones ópticas no modeladas, o por el error de los parámetros que entren en el modelo. Por tanto, el resultado no será una traslación "perfectamente rígida". Por ejemplo, en general se cumplirá que $\left|\overrightarrow{(p_i)_{RK}(p_j)_{RK}}\right| \cong \left|\overrightarrow{(p_i)_{R1}(p_j)_{R1}}\right|$ a pesar de que deberían ser iguales, por ser R1 y las RK rectangulares.

Retomando la perspectiva del ajuste de poliedros, esto puede entenderse como que los dos poliedros no encajarán perfectamente. Sin embargo, la elaboración de un método que determinara un ajuste de los puntos de ambos poliedros por, por ejemplo, mínimo error cuadrático, supondría una complicación del problema que no parece merecer la pena considerando lo concerniente a la precisión en los objetivos marcados. Los resultados experimentales corroboran que el error es despreciable, con valores para el determinante de la matriz de cambio de base asociada a M_k de 1 ± 10^{-5} y, de forma similar, un resultado para la multiplicación por su traspuesta muy cercano a la identidad.

Fase 1 de calibrado: Extracción de las primitivas 2d del patrón de calibración [4]

El patrón de calibrado debe ser un objeto físico con 4 puntos cuyas distancias relativas sean conocidas y cuyas propiedades geométricas y superficiales favorezcan la identificación precisa de esos puntos. Por cumplir estos requisitos y además por su facilidad de obtención, se ha elegido una hoja DIN-A4 pegada sobre una superficie plana y oscura, que sobresalga unos centímetros, lo suficiente para crear un marco de alto contraste, como una carpeta negra. En cualquier caso, ésta es una decisión que afecta principalmente a este módulo del sistema, de modo que un cambio sólo requeriría unos pequeños ajustes en la fase 2 y 3 del calibrado, o simplemente en un fichero de configuración externo si se emplea una técnica de *softcoding* adecuada en la implementación. Así, se podría cambiar fácilmente y calibrarse el sistema con cualquier otro tipo de patrón real que cumpliera las anteriores condiciones, no necesariamente con los puntos coplanarios, lo cual añade versatilidad al sistema. Los sistemas basados en la técnica descrita por Duda y Hart o la de Tsai necesitan patrones con muchos puntos, en algunos casos necesariamente coplanarios, para lo cual en aplicaciones prácticas es común emplear un tablero de ajedrez de dimensiones conocidas. Sucede lo mismo en el caso de implementaciones como la librería OpenCV.

A continuación se describirán los algoritmos de procesado de imagen empleados para detectar las primitivas en el objeto patrón elegido.

Dado que el sistema está pensado para permitir movimientos amplios por parte del usuario, las cámaras han de estar a una distancia de entre 1m y 3m del centro de la zona localizable (i.e. la que captan al menos dos cámaras, que pueden ser las únicas si el sistema no cumple el objetivo adicional OA2). Ahora hay que recordar que existen dos variantes de la fase 2 de calibrado, por lo que esta fase primera deberá ofrecer una salida adecuada para ambas.

Como se explicó en el apartado correspondiente, en la variante para la resolución del sistema 5x5 cada instancia de la cadena de llamadas Fase1->Fase2 es individual para cada cámara, siendo el propósito resolver de forma más precisa la distancia focal de ésta. En este caso, el patrón se acerca a la cámara correspondiente, y por tanto es esperable que este ocupe una zona de la imagen más bien global. Este paso es prescindible para las cámaras cuya distancia focal equivalente se conoce, pero aún así debe ser considerado.

En la variante para la resolución del sistema 3x3, a diferencia de en la anterior, el patrón de calibrado debe ser localizado simultáneamente por todas las cámaras para poder llevar a cabo lo explicado en el apartado anterior, por lo que debe estar en la zona localizable común, y por tanto ocupará una zona de la imagen más local que global, como se puede apreciar en la figura (11.1(a)).

[4] En este capítulo, así como en el de "Fase 1 de localización en tiempo real", se emplea terminología específica del campo de tratamiento digital de imagen, que aunque se reduce rápidamente a conceptos matemáticos generales, puede requerir una introducción para el lector no familiarizado. Esta puede encontrarse, entre otra mucha bibliografía, en *Digital Image Processing* (González, Woods, 2008). Para agilizar la tarea, a continuación se hace una lista de términos con su correspondiente localización en dicha fuente: Transformada de Hough [p.733], Transformaciones morfológicas binarias (erosión, dilatación) [p. 630], Umbralización [p.738], Teoría de color (espacio YUV, luminancia, crominancia) [p.394], Métodos de extracción de bordes (Operador de Sobel) [p.167,p.708].

Esto nos lleva a una conclusión: la algoritmia de la fase debe ser flexible frente al tamaño relativo del patrón en la imagen. Esto a su vez nos lleva a tomar la decisión de diseño consistente en dividir la fase en dos subfases claramente diferenciadas: una de localización del patrón en la escena, y otra de localización de los puntos en el patrón.

(a)
Entrada de la primera subfase: imagen global, donde el patrón es local.

(b)
Salida de la primera subfase y entrada de la segunda subfase: imagen local, donde el patrón es global.

(c)
Salida de la segunda subfase: coordenadas de los puntos.

Ejemplo de entrada y salida de la fase 1 de calibrado y de sus dos subfases.

Figura 11.1

SUBFASE 1: Localización del patrón en la escena

La hoja de papel sobre un marco oscuro ofrece características notables de luminancia en el centro y de contraste en los laterales. Sin embargo, estas características por sí solas no hacen suficientemente obvia la identificación en una escena general. Fácilmente se dan objetos con la misma o mayor luminancia que la superficie del papel, sobre todo en ciertas condiciones de iluminación. Acudir a patrones geométricos, como bordes rectos, o a información de color, como la disponible al emplear una hoja de un color específico, sufre del mismo problema: si se desea una especial robustez del algoritmo para entornos muy generales, no es difícil que estas características se repitan fuera del patrón de calibrado. Esta posibilidad de error se reduce a unas cotas más aceptables si se reduce el patrón de búsqueda a un modelo geométrico más específico, como por ejemplo el de cuadrilátero de interior claro sobre fondo oscuro. Sin embargo, el coste computacional para el reconocimiento de patrones tan elaborados sobre toda la imagen es demasiado elevado: por ejemplo, una transformada de Hough orientada a cuadriláteros implica una búsqueda sobre un espacio de 8 dimensiones (2 por cada punto) que se hace inabordable incluso para resoluciones muy bajas de imagen. Este tipo de búsqueda pertenece realmente a la segunda subfase, ya que permite localizar directamente las esquinas, que son las primitivas.

Por consiguiente, hay que emplear un tipo de información que distinga notablemente el patrón del fondo, y que el patrón de por sí no posea. Aunque son posibles muchas opciones que pasan por complicar el patrón con elementos que añadan características

muy llamativas sobre el entorno, una información muy sencilla de extraer y que implica pocas instrucciones hacia el usuario es el movimiento. Si se asume que la escena va a ser estática, lo cual parece posible para una gran mayoría de situaciones, el valor absoluto de la diferencia entre dos fotogramas separados por un intervalo de alrededor de un segundo, habiéndosele indicado al usuario que realice un movimiento sobre el patrón, puede ser suficiente.

(a) Diferencia

(b) Diferencia umbralizada

$$D(t) = \left| Y(t) - Y(t-1) \right| \qquad D'(t) = \begin{cases} D(t) & D(t) > thresold \\ 0 & D(t) < thresold \end{cases}$$

Figura 11.2

El movimiento más indicado es el de rotación de unos pocos grados (20º aprox.) con respecto al eje perpendicular al plano del patrón, y centrado en éste. Dado que el patrón consiste en un rectángulo de alta luminosidad sobre un marco oscuro, la imagen diferencia poseerá un aspecto similar al de la figura (11.2(a)), manteniendo la marca de la rotación unos valores de diferencia bastante altos.

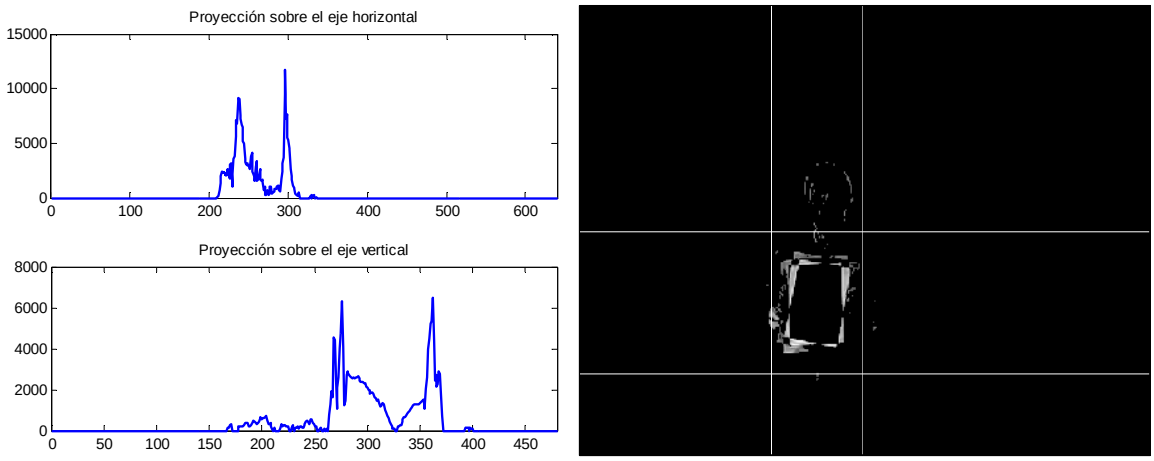
Debido al ruido de fondo en el sensor, aparecerá uno similar en la imagen diferencia, pero por lo mencionado sobre los valores elevados de la marca de rotación, es posible tratar la imagen como en la figura (11.2(b)) con un valor de umbral que no borre la marca pero elimine o al menos reduzca el ruido para cualquier imagen típica de un sensor CCD.

Este tratamiento favorece el siguiente paso: localizar un rectángulo contenido en la imagen que contenga la marca de rotación y por tanto la imagen del patrón en cualquiera de las dos imágenes consecutivas que se han empleado para su elaboración. Este rectángulo definirá los límites de la subimagen que se pasará a la siguiente subfase.

El método elegido proyecta sobre la horizontal los valores de $D'(t)$, y detecta sobre esa señal resultante los límites horizontales. El método empleado busca, como siempre, unos resultados lo más independientes posible de las características puntuales de cada imagen, y consiste en recorrer la señal tanto desde el extremo izquierdo como el derecho hasta encontrar, en cada caso, el primer valor igual o mayor que el valor

máximo de la señal dividido por una constante. En la práctica se ha demostrado que un valor 3 para la constante suele dar buenos resultados.

Posteriormente, se repite el proceso con una proyección sobre la vertical de la subimagen resultante entre los dos límites horizontales hallados, y así se obtienen los límites verticales. Ya se tiene la subimagen.



(a) Proyecciones

(b) Imagen global con selección

Figura 11.3

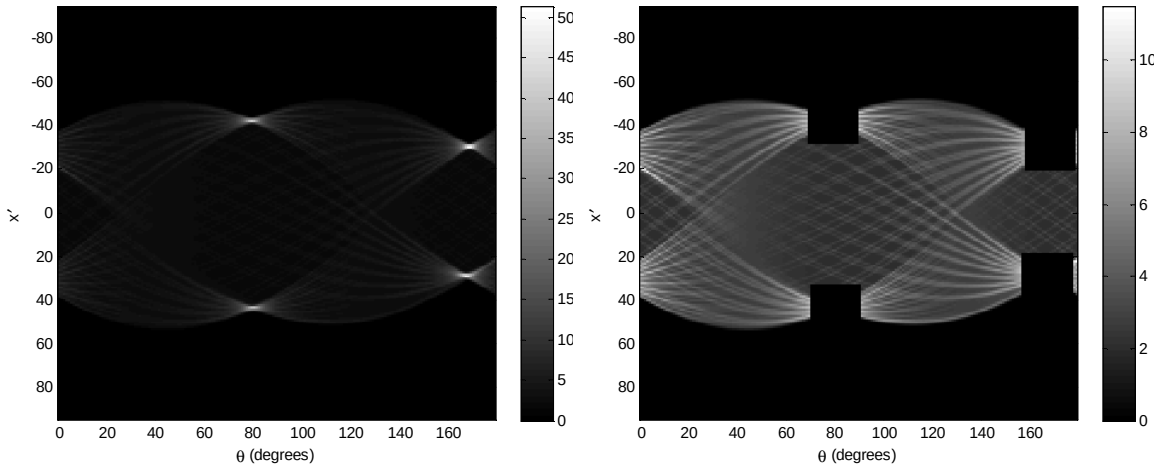
SUBFASE 2: Localización de los puntos en el patrón.

Ahora se trabaja con una subimagen que presumiblemente abarca el patrón de calibrado y sus alrededores, como la que se puede observar en la figura (11.1(b)). Hay que señalar que esta subimagen no se remuestrea ni interpola, tanto por motivos de carga computacional como de precisión, y por tanto el algoritmo ha de poder trabajar en resoluciones arbitrarias. Para lograr el objetivo, es decir, localizar las esquinas de la hoja, se ha elegido aplicar una extracción de bordes y emplear una transformada de Hough para detectar individualmente las 4 rectas que forman la frontera entre hoja y marco, y finalmente las 4 intersecciones. Si bien hay métodos más directos, como operadores basados en segundas derivadas (discretas) que producen máximos en el contacto con esquinas, estos provocarían fácilmente falsas alarmas, por la naturaleza marcadamente local de estos operadores. Además, las esquinas generadas en imágenes reales no son ideales, y pueden presentar fácilmente cierto radio de curvatura, de unos pocos píxeles pero variable. Esto obligaría probablemente a emplear baterías de detectores de esquinas escalados, lo cual añade mucha complicación.

La transformada de Hough tampoco es el método más ligero computacionalmente, pero es bastante robusto debido a que aprovecha la información a lo largo de todas las líneas para crear una recta 'media'. Rigurosamente no es la recta media sino la modal; si en la transformada de Hough se seleccionan los máximos locales (que es el método más típico y el seguido aquí), el resultado es la recta que interseca con más píxeles pertenecientes al borde. Pero el efecto es similar, en el sentido de que si las rectas están ligeramente curvadas por efecto de distorsión de lente, irregularidades en la superficie

plana que sostiene la hoja de papel, o porque la instantánea está ligeramente movida, el algoritmo arrojará una solución en cuyo valor influirá el valor de todos los puntos de la frontera, no un subconjunto, y el resultado estará 'equilibrado' (hay casos sintéticos en los que la recta modal y la recta que minimiza distancias cuadráticas difieren mucho, pero estos no son los típicos que se dan en imágenes naturales). Incluso si hay cortes en las rectas o las mismas esquinas no han superado el umbral del detector de bordes, el método sigue funcionando mientras un conjunto de puntos suficiente se mantenga fiel al borde. Asimismo, y precisamente por basarse en la moda y no en la media, el método es inmune a outliers, que vienen dados por desviaciones espurias en la recta. En la práctica, estas irregularidades no son infrecuentes, y de forma similar a los cortes, se dan típicamente por falta de contraste entre hoja y marco, debido a condiciones especiales de iluminación, o materiales que sujetan el papel al marco.

Los máximos buscados sobre la transformada de Hough son los 4 principales máximos locales, no globales. Por tanto, se emplea una subrutina que busca el máximo global en la transformada, lo guarda, y sobreescribe con 0 un entorno dentro del cual no debería haber otro máximo relacionado con otra esquina del patrón, pero sí puede haber otro relacionado con la misma (P.ej: $[\pm 20º, \pm 10px]$), y se itera sobre dicha subrutina 4 veces.



(a) Transformada de Hough de *solamente* los cuatro bordes generados por el patrón.

(b) Resultado tras el algorimo de detección. (Margen dinámico de la imagen alterado)

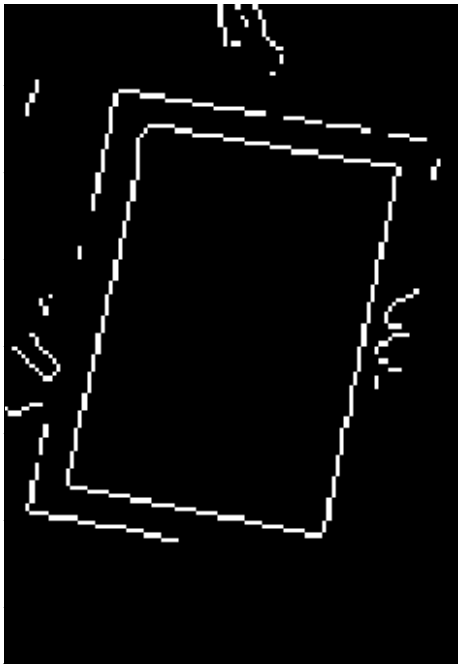
Figura 11.4

Finalmente, se resuelven los cortes que dan las rectas representadas por los puntos máximos extraídos de la transformada, y de las 6 soluciones se emplean las 4 que caen dentro de la subimagen. Se podría emplear como función discriminante la luminancia media de las regiones abarcadas por todos los subconjuntos de 4 puntos, pero no es necesario porque para las distancias focales típicas, el tamaño del patrón y las distancias típicas entre este y las cámaras, la diferencia de pendiente entre rectas correspondientes a lados paralelos del patrón es suficientemente pequeña, y por tanto sus cortes son suficientemente lejanos con respecto a los límites de la subimagen. Eso sí, se hace necesario emplear una pequeña distorsión de 1º para evitar problemas de condicionamiento cuando dos puntos quedan alineados a nivel píxel y por tanto su pendiente es infinita. Sería posible emplear un sistema de referencia alternativo para

este caso, pero la insignificancia del error producido por la distorsión ocasional de 1º justifica el empleo de la opción sencilla.

Esta última etapa de resolución de cortes en el plano podría evitarse si la transformada de Hough empleada fuese la generalizada a cuadriláteros, que ya se mencionó en la discusión de la subfase anterior, y cuyo resultado consiste ya directamente en los cuatro puntos buscados. Sin embargo, aunque la subimagen debe tener dimensiones menores que la imagen, es fácil que no sea lo suficientemente pequeña como para que el método se resuelva en un tiempo razonable, y emplear remuestreo tendría efectos negativos sobre la precisión de los puntos obtenidos. En cambio, el método de la transformada de Hough de rectas traba sobre 2 dimensiones en lugar de 8, y la posterior intersección de rectas en el plano es computacionalmente muy ligera frente a las operaciones de procesado de mapas de bits, de modo que para cualquier situación, el conjunto es mucho más ligero. Experimentalmente se ha comprobado que para resoluciones grandes de la subimagen (120Kpx.), procesadores típicos y en un lenguaje interpretado, se toma del orden de un segundo por imagen.

Llegados a este punto ya se tiene un prototipo funcional para la segunda subfase del extractor de primitivas bidimensionales del patrón de calibrado. Sin embargo, en la práctica es fácil que falle. La causa es que aunque la subimagen omite gran parte del fondo, aún puede haber otras líneas rectas que den valores en el espacio de Hough mayores que alguna de las 4 buscadas. En concreto, es fácil que si el fondo tras el patrón no es oscuro como este, se genere un segundo cuadrilátero externo al que forma el papel sobre el marco. Esto puede apreciarse en la figura (11.5), donde a la subimagen se le ha aplicado como extractor de bordes un operador de Sobel, muy empleado para este propósito. En situaciones así, lo más probable es que alguno, si no varios, de los bordes exteriores, arroje picos mayores en el espacio de Hough que alguno de los interiores.



Bordes extraídos de la subimagen en un caso típico.

Figura 11.5

Considerando el trabajo de la primera subfase, puede asumirse que el centro de la subimagen coincidirá con un punto interior de la hoja del patrón de calibrado. Por tanto, una posible solución sería binarizar la imagen con un umbral típicamente intermedio y quedarse con la región conexa a dicho punto, que si se trata de la correspondiente a la hoja dentro del marco, sólo dará lugar a los bordes entre ambos. Aunque el hecho de que se elija un patrón formado por una hoja blanca sobre un marco negro facilita el empleo de un umbral de binarización 'típicamente intermedio', la experiencia ha demostrado que las variaciones de iluminación y, quizá más aún, de tipo de cámara, hacen que el empleo de un valor único para todos los posibles casos sea propenso a fallos fuera de las condiciones que se han empleado para determinarlo. Esto se agrava si

algún objeto (por ejemplo el material empleado para fijar la hoja sobre el marco) cruza este último y tiene una luminosidad alta, ya que facilita que la hoja quede conectada con el fondo en la imagen umbralizada.

Por tanto, se ha empleado un algoritmo para detectar el borde de la hoja con el marco independientemente de los valores de luminancia de ambos. Considérese la función:

$$\Delta S(t) = \left| conreg(Y > (2^n - t), size(Y)/2) \right| - \left| conreg(Y > (2^n - t + 1), size(Y)/2) \right| \quad (11.1)$$
$$0 \le t < 2^n$$

donde $conreg(B, p)$ devuelve el conjunto de puntos 4-conectados a p en la imagen binaria B, $size(Y)/2$ es el punto medio de la imagen de luminancia Y de n bits, y el operador > representa binarización por umbral simple.

El comportamiento esperado se describe en la siguiente tabla:

$0 \le t < l_1$ $2^n - l_1 = Y(size(Y)/2)$	$\forall t, S(t) = 0 \Rightarrow \Delta S(t) = 0$. Porque el punto de conexión está por debajo del umbral
$l_1 \le t < l_2$ $2^n - l_2 = \min(Y(P_H))$ $P_H = \{p / p \in Hoja.\}$	$S(t)$ crece rápidamente $\Rightarrow \Delta S(t) = \uparrow\uparrow$ Esto se debe a que, por ser la luminancia a lo largo de toda la hoja de valores similares y sin discontinuidades espaciales bruscas, a cada paso en el umbral, se incorpora un área extensa a la región conexa.
$l_2 \le t < l_3$ $2^n - l_3 = \max(Y(P_M))$ $P_M = \{p / p \in Marco:\}$	$S(t)$ crece poco $\Rightarrow \Delta S(t) \approx 0$ Esto se debe a la discontinuidad a lo largo de la frontera que separa la hoja del marco. Esta discontinuidad debe ser, valga la contradicción, continua, o de lo contrario el algoritmo conectará la región con otras zonas luminosas del exterior del patrón.
$l_3 \le t < 2^n$	$S(t)$ crece rápidamente $\Rightarrow \Delta S(t) = \uparrow\uparrow\uparrow$ Por ser la luminancia del marco también homogénea, una vez que $2^n - t$ queda por debajo del máximo valor del marco, rápidamente sucederá para otros muchos puntos del marco y la umbralización conectará el interior del patrón, la hoja, con el fondo que rodea el marco. Dado que $2^n - t$ ya poseerá valores bajos, del orden de los del marco oscuro, se producirá una rápida expansión por toda la imagen hasta que la región conexa constituya toda la imagen, ya con $2^n - t = 0$.

Tabla 11.1

Entonces, para hallar la frontera entre hoja y marco, que es la que a través de la transformada de Hough dará las esquinas de la hoja, puede centrarse la tarea en encontrar, dada una imagen, un valor de *t* comprendido en el tercer intervalo (l_2, l_3), y aplicar entonces un extractor de bordes a $conreg(Y > (2^n - t), size(Y)/2)$. La descripción misma del intervalo (l_2, l_3) nos muestra que si la frontera entre hoja y marco es suficientemente nítida (que ha de serlo si la cámara está adecuadamente enfocada, lo cual se presupone), no importa mucho el valor de *t* que se elija dentro de él, pues la diferencia resultante puede ser a lo sumo la de la expansión de la región conexa en pocos píxeles. Este valor se puede encontrar detectando los niveles bajos de $\Delta S(t)$, pero nuevamente esto hay que hacerlo mediante un método independiente de las medidas absolutas en píxeles del patrón en la imagen, que dependerán de la distancia y la orientación de este con respecto a la cámara, así como de la distancia focal de esta. Por ello, se puede tomar como referente el perímetro del patrón en cada momento, y determinar una constante *c* tal que:

$$\begin{aligned} \Delta S(t) &> c \cdot P(t) \qquad & l_1 < t < l_2 \\ \Delta S(t) &< c \cdot P(t) \qquad & l_2 < t < l_3 \end{aligned} \tag{11.2}$$

Una vez se tenga esa constante, sólo hay que iterar sobre *t* creciente, y quedarse con el primer valor de *t* para el que $\Delta S(t) < c \cdot P(t)$ y $S(t) > 0$.

La ventaja de este método es que es invariante sobre los niveles de iluminación de la hoja y del marco. La constante *c* no depende de estas, sino de la brusquedad de los cambios de luminancia dentro de la hoja y en la frontera con el marco.

El valor del perímetro de una región puede obtenerse del cardinal (número de píxeles con valor 1 en una imagen binaria) de la diferencia entre la dilatación de la imagen con una máscara 3x3 y la propia imagen. Formalmente:

$$P(t) = \left|(R \oplus S_3) - R\right|$$

$$R = conreg(Y > (2^n - t), size(Y)/2) \qquad S_3 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{11.3}$$

Sin embargo, esto es bastante costoso computacionalmente, especialmente si se tiene en cuenta que se evalúa dentro de un bucle. Por ello, en la implementación se ha optado por una aproximación más ligera, válida para regiones conexas, sin agujeros, de frontera regular y dimensiones vertical y horizontal similares, que son propiedades todas esperables para *R*.

$$P(t) = \sqrt{|R|} \tag{11.4}$$

Si la frontera está desenfocada, el valor de *c* deberá ser más alto, mientras que si existen discontinuidades en la hoja como por ejemplo una región grande en sombra debida a un obstáculo entre ella y la fuente de luz, el valor de *c* deberá ser lo suficientemente bajo

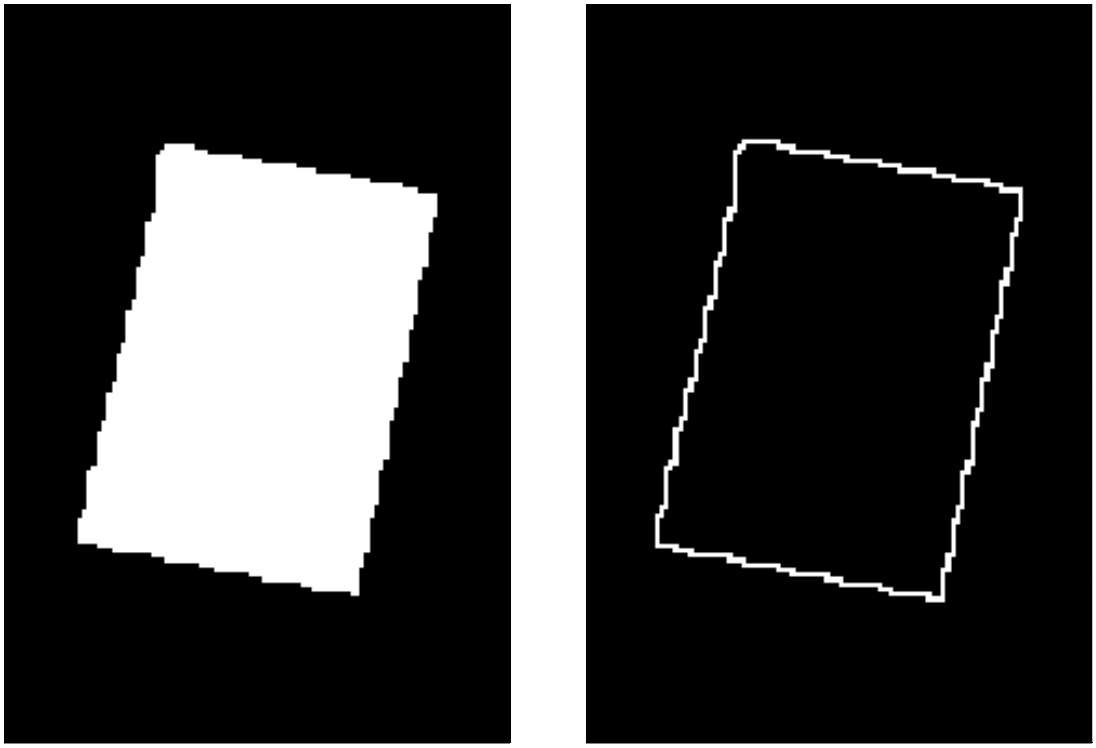
como para que el algoritmo no pare en el borde de la sombra. Pero en general, para la mayoría de imágenes típicas, el resultado es el adecuado empleando el mismo valor *c*. En el desarrollo de la implementación se ha comprobado que para condiciones típicas, con una imagen no desenfocada, un valor de *c=1* es apropiado.

Para evitar que los bloques de la compresión que aplican algunas cámaras puedan crear falsos positivos, al haber fronteras de más de un nivel de luminancia a lo largo de la degradación progresiva en la superficie de la hoja, la iteración se hace con saltos mayores que la unidad. Otra opción más costosa computacionalmente sería aplicar un filtro paso bajo previamente. Aún así, el valor del salto puede ser suficientemente grande como para que no haya pérdidas fácilmente. Para luminancia de profundidad 8 bits, un valor de salto de 5 ha demostrado ofrecer buenos resultados en este sentido, a la vez que no es susceptible a 'saltarse' el verdadero umbral.

Una vez detectado el umbral *t*, se pueden extraer los bordes mediante un operador de Sobel. También se puede aprovechar que se trata de una imagen binaria, y emplear una dilatación similar a la de la ecuación (11.3), con la misma definición para *R* y S_3:

$$B = (R \oplus S_3) - R \tag{11.5}$$

En este punto se obtiene una imagen semejante a la de la figura (11.6(b)), y se podría calcular sobre ella una Transformada de Hough como la de la figura (11.4), procediendo con los pasos descritos con anterioridad.



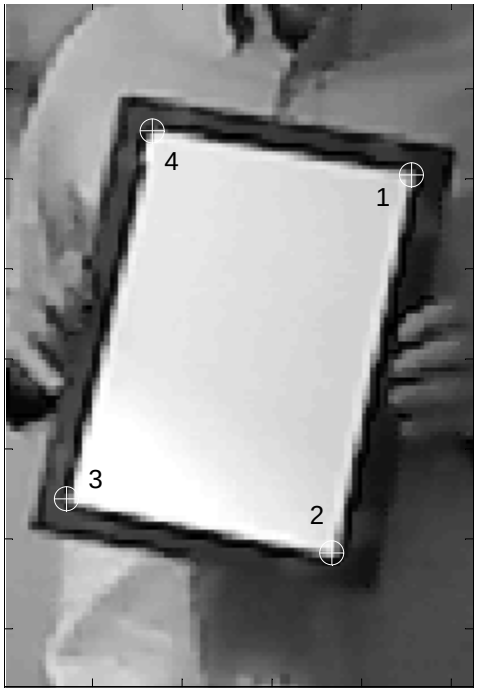
(a)
Región conexa detectada tras la parada del algoritmo.

(b)
Bordes extraídos mediante la ecuación (11.5)

Figura 11.6

Si bien la ecuación (11.5), frente a otros métodos basados en la convolución con un operador de orden impar, como el de Sobel, produce un borde exterior menos centrado en la frontera real de la figura (11.6 (a)), este efecto es de sólo ½ píxel de desplazamiento para los bordes, y para la mayoría de ángulos en las esquinas, de no más de 1 píxel en estas. Pero teniendo en cuenta que el valor de *c* apropiado va a ser suficientemente alto como para asegurar que en imágenes algo desenfocadas la condición sobre el bucle también salte, en general va a producir una respuesta algo temprana, sobre lo cual esta desviación actúa de forma contraria. Los resultados, como en la figura (11.7), son satisfactorios.

Finalmente, los puntos se ordenan por el orden de las agujas del reloj con respecto a su centro geométrico, que es lo convenido con los otros módulos, y permite que se asocie adecuadamente cada punto encontrado en la imagen con el correspondiente punto tridimensional del patrón de calibrado, y la información almacenada sobre las distancias mutuas entre puntos tridimensionales sea coherente con la imagen. Esto obliga a que el patrón de calibrado deba ser usado también en una posición convenida, concretamente la hoja debe estar en posición suficientemente vertical en las imágenes de todas las cámaras. Esto impide que las cámaras difieran mucho en su ángulo *roll*, lo cual no parece una imposición muy severa si se tiene en cuenta que para apuntar el eje óptico a cualquier dirección, sólo es necesario manejar los ángulos *yaw* y *pitch*.



Ejemplo de salida de la implementación de la segunda fase.

Figura 11.7

Por último, se transforman las coordenadas de los puntos bidimensionales al sistema de referencia de la imagen completa, con el origen en el centro de la imagen, de forma que puedan ser empleadas por la fase 2 de calibrado.

Fase 2 de localización en tiempo real: **Cálculo de la posición tridimensional del punto localizable.**

Tras el calibrado, el sistema dispone de $\{f_1, \ldots, f_n\}$, $\{M_1, \ldots, M_n\}$ con $M_1=I$, y esta información se supone válida hasta nuevo calibrado.

En cada iteración, la fase 1 de localización en tiempo real halla $\{(p_{p1})_{R1}, \ldots, (p_{pn})_{RN}\}$, información de la que hace uso este módulo. A diferencia de en la discusión de la fase 2 de calibrado, p_{pi} hace referencia a la proyección del punto p en la cámara i, y no a la proyección del punto p_i en una única cámara. Dado que, a diferencia de en la etapa de calibrado, la labor de localización de distintos puntos es independiente (y además parte de un objetivo opcional) se trabajará con la localización de un solo punto p, simplificando así la notación. La notación indica además el hecho de que cada p_{pi} es devuelto en función del sistema de referencia Ri, el de su cámara.

Debido a las distorsiones no modeladas o modeladas con parámetros con margen de error, p como tal es indeterminable. Por tanto, se hablará de la localización de $\tilde{p}$, una aproximación según los datos disponibles. A continuación se expondrán dos métodos, ambos válidos para el caso de 2 y más cámaras, y que lidian con el problema de la incoherencia introducida por los errores mencionados.

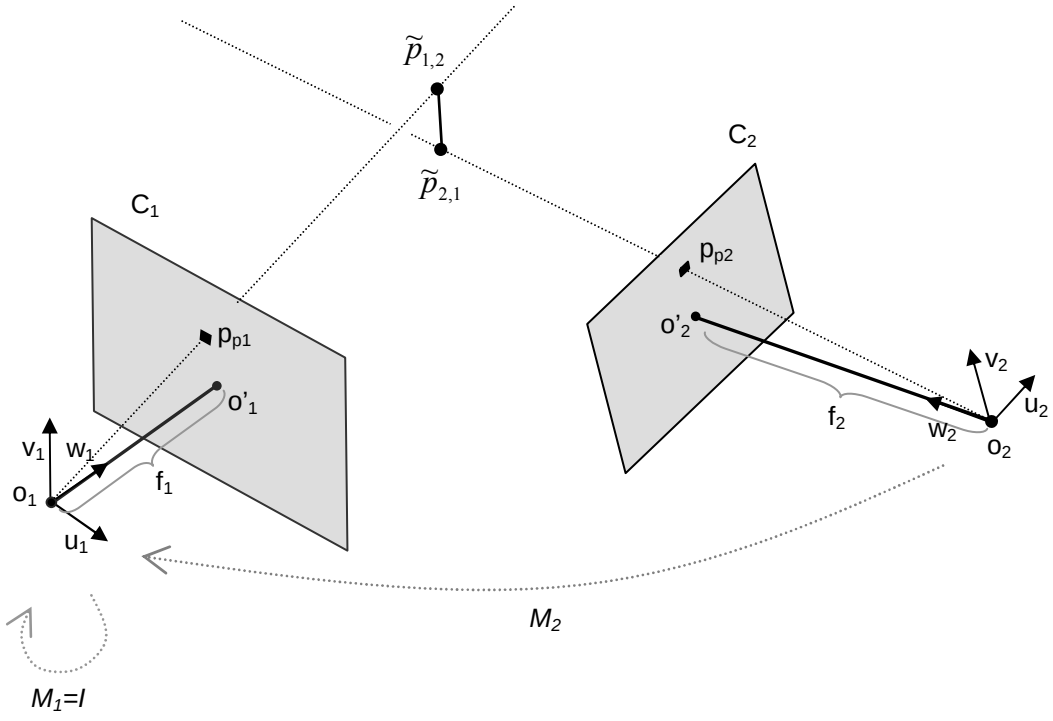


Figura 12.1

Nótese que en este caso no se trata de la misma cámara en distintos sistemas de referencia, como podría parecer por analogía con la figura 1.3 del apartado de descripción del modelo pinhole, sino de dos cámaras distintas, una de ellas (C1) cuyo sistema de referencia asociado es el que se emplea como global. Para hacer énfasis en ello, se ha dibujado un ejemplo en el que C2 tiene un tamaño de sensor (resolución según lo convenido) y una distancia focal distintas a C1.

Ambos métodos comparten una metodología inicial. El sistema convierte $(p_{pi})_{Ri}$ a $(p_{pi})_{R1}$, es decir, cada punto proyectado al sistema de referencia global. De forma similar, aunque esto no se tiene por qué hacer en cada iteración, se dispone de $(o_i)_{R1}$, que se extrae de forma trivial de M_i . De esta forma, en cada iteración se dispondrá de los datos definitorios de la recta $\overline{o_i p_{pi}}$ en el sistema de referencia global. En adelante se trabajará con él por defecto.

El primer método es el más obvio para el caso de dos cámaras. Teniendo en cuenta que por los errores, las rectas $\overline{o_i p_{pi}}$ no se cortarán sino que se cruzarán, el objetivo es encontrar el punto $\tilde{p}$ que minimice la distancia a todas ellas. En el caso de dos cámaras, sólo han de calcularse el punto $\tilde{p}_{1,2}$, que es el contenido en la recta $\overline{o_1 p_{p1}}$ que minimiza la distancia a $\overline{o_2 p_{p2}}$, y el punto $\tilde{p}_{2,1}$, cuyo caso es el inverso. Esto se puede hacer con una sencilla fórmula de producto escalar igual a cero. $\tilde{p}$ se puede definir entonces como la media de $\tilde{p}_{1,2}$ y $\tilde{p}_{2,1}$, asegurándose la condición de Mínimo Error Cuadrático (MSE) con respecto a las rectas. Para el caso de *n* cámaras, el método se puede generalizar, hallando para cada cámara *k* los puntos $\{\tilde{p}_{k,m} / m \in [1,n] - \{k\}\}$. Esto define una nube de puntos cuyo centro de masas puede considerarse una buena aproximación, aunque sin garantía matemática (al menos a simple vista) de cumplir la condición de MSE con respecto a todas las rectas.

Una opción interesante sería la de poder asignar más peso a la información proporcionada por determinadas cámaras, según su fiabilidad estimada. Para dos cámaras, esto puede hacerse, por ejemplo, en función de la incoherencia obtenida en su calibración, o la resolución de la imagen. Si se espera que la distorsión radial se la principal fuente de error, también puede asignarse menos peso a la información dada por una primitiva bidimensional cercana a los bordes de la imagen, ya que el efecto de esta distorsión aumenta en función del cuadrado a la distancia al centro de la imagen. Para más de dos cámaras, puede emplearse la distancia de cruce entre pares de rectas $\overline{o_i p_{pi}}, \overline{o_j p_{pj}}$ para asignar subconjuntos de cámaras con coherencia mutua mayor, y por tanto presumiblemente más fiables. En este método esto se puede conseguir haciendo una media ponderada de los puntos $\tilde{p}_{k,m}$ asignando mayor peso a los relacionados con las cámaras mas fiables (habría que considerar que cada $\tilde{p}_{k,m}$ está asociado tanto a la cámara *k* como a la *m*, aunque es mayor el vínculo con la *k*).

En general, el método descrito es conceptualmente sencillo, ya que se basa en la generalización directa del método más obvio para dos cámaras, aunque para un hipotético número de cámaras muy alto puede ser ineficiente, por la explosión combinatoria.

El segundo método desarrollado se basa en una resolución analítica del $\tilde{p}$ como solución con mínimo error cuadrático medio. Formalmente:

$$\tilde{p} = \arg\min_p \bar{d}(p) \qquad \bar{d}(p) = \sum_{i=1}^{n} d(p, \overline{o_i p_{pi}})^2 \tag{12.1}$$

Emplear error cuadrático permite resolver el mínimo mediante un sistema de ecuaciones lineales, que es el que resulta de igualar el gradiente de la función a cero. La gran ventaja es que el sumatorio, en el cual cada sumando representa la distancia a la recta $\overline{o_i p_{pi}}$ dada por la cámara i , puede salir fuera de la derivada, haciendo que la construcción del sistema para las n cámaras no sea más que la suma de las n matrices y n vectores de términos independientes, consiguiendo así una complejidad asintótica $\Theta(n)$, siendo n el número de cámaras, frente a la $\Theta(n!)$ del primer método. Formalmente:

$$\bar{d}(p) = \sum_{i=1}^{n} d(p, \overline{o_i p_{pi}})^2 = \sum_{i=1}^{n} \left(\frac{\left\| \overline{o_i p} \times \overline{o_i p_{pi}} \right\|}{\left\| \overline{o_i p_{pi}} \right\|} \right)^2 =$$

$$= \sum_{i=1}^{n} \frac{\begin{array}{l} ((\{o_i\}_2 - \{p_{pi}\}_2)(\{o_i\}_1 - \{p\}_1) - (\{o_i\}_1 - \{p_{pi}\}_1)(\{o_i\}_2 - \{p\}_2))^2 + \\ ((\{o_i\}_3 - \{p_{pi}\}_3)(\{o_i\}_1 - \{p\}_1) - (\{o_i\}_1 - \{p_{pi}\}_1)(\{o_i\}_3 - \{p\}_3))^2 + \\ ((\{o_i\}_3 - \{p_{pi}\}_3)(\{o_i\}_2 - \{p\}_2) - (\{o_i\}_2 - \{p_{pi}\}_2)(\{o_i\}_3 - \{p\}_3))^2 \end{array}}{(\{p_{pi}\}_1 - \{o_i\}_1)^2 + (\{p_{pi}\}_2 - \{o_i\}_2)^2 + (\{p_{pi}\}_3 - \{o_i\}_3)^2} \tag{12.2}$$

Cada elemento del sumatorio es una función cuadrática $\Re^3 \to \Re$ con una recta en el espacio origen cuyo valor es mínimo en la función e igual a cero. Esta recta es $\overline{o_i p_{pi}}$ para la función asociada a la cámara *i*. Para dos o más cámaras, que es el caso considerado, y teniendo en cuenta que $\overline{o_i p_{pi}} \neq \overline{o_j p_{pj}} \ \forall i \neq j$, habrá un solo mínimo absoluto, que será igual a cero si las rectas se cortan, o no si sólo se cruzan. En cualquier caso, considerando la naturaleza cuadrática de las funciones, será también el único mínimo local, por lo que para hallarlo sólo hay que igualar el gradiente a cero.

$$\begin{cases} \dfrac{\partial \bar{d}}{\partial \{p\}_1} = 0 \\ \dfrac{\partial \bar{d}}{\partial \{p\}_2} = 0 \\ \dfrac{\partial \bar{d}}{\partial \{p\}_3} = 0 \end{cases} \tag{12.3}$$

También por ser cuadráticas, se tiene que las derivadas parciales forman un sistema lineal. Para la primera derivada parcial, el desarrollo es el siguiente:

$$\frac{\partial \bar{d}}{\partial \{p\}_1} = \sum_{i=1}^{n} \frac{\partial \begin{pmatrix} (((\{o_i\}_2 - \{p_{pi}\}_2)(\{o_i\}_1 - \{p\}_1) - (\{o_i\}_1 - \{p_{pi}\}_1)(\{o_i\}_2 - \{p\}_2))^2 + \\ (((\{o_i\}_3 - \{p_{pi}\}_3)(\{o_i\}_1 - \{p\}_1) - (\{o_i\}_1 - \{p_{pi}\}_1)(\{o_i\}_3 - \{p\}_3))^2 + \\ (((\{o_i\}_3 - \{p_{pi}\}_3)(\{o_i\}_2 - \{p\}_2) - (\{o_i\}_2 - \{p_{pi}\}_2)(\{o_i\}_3 - \{p\}_3))^2 \end{pmatrix}}{\partial \{p\}_1}}{(\{p_{pi}\}_1 - \{o_i\}_1)^2 + (\{p_{pi}\}_2 - \{o_i\}_2)^2 + (\{p_{pi}\}_3 - \{o_i\}_3)^2} =$$

$$= \sum_{i=1}^{n} \frac{a_{1,1} \cdot \{p\}_1 + a_{1,2} \cdot \{p\}_2 + a_{1,3} \cdot \{p\}_3 + b_1}{(\{p_{pi}\}_1 - \{o_i\}_1)^2 + (\{p_{pi}\}_2 - \{o_i\}_2)^2 + (\{p_{pi}\}_3 - \{o_i\}_3)^2} = 0$$

con:

$$\begin{aligned} a_{1,1} &= 2 \cdot ((\{o_i\}_2 - \{p_{pi}\}_2)^2 + (\{o_i\}_3 - \{p_{pi}\}_3)^2) \\ a_{1,2} &= 2 \cdot (\{o_i\}_1 - \{p_{pi}\}_1) \cdot (\{o_i\}_2 - \{p_{pi}\}_2) \\ a_{1,3} &= 2 \cdot (\{o_i\}_1 - \{p_{pi}\}_1) \cdot (\{o_i\}_3 - \{p_{pi}\}_3) \\ b_1 &= 2 \cdot ((\{o_i\}_2 (\{o_i\}_1 - \{p_{pi}\}_1)) - (\{o_i\}_1 (\{o_i\}_2 - \{p_{pi}\}_2)^2) + \\ &+ (\{o_i\}_3 (\{o_i\}_1 - \{p_{pi}\}_1)) - (\{o_i\}_1 (\{o_i\}_3 - \{p_{pi}\}_3)^2)) \end{aligned}$$

El procedimiento es análogo para la segunda y tercera derivadas parciales, obteniendo así $a_{2,1}, a_{2,2}, a_{2,3}, b_2, a_{3,1}, a_{3,2}, a_{3,3}, b_3$, y formando para cada cámara *i* una matriz A_i 3x3 y un vector $B_i \in \Re^3$ de términos independientes. El sistema total se obtiene dividiendo por el denominador, distinto para cada cámara, y calculando el sumatorio. El sistema final queda como:

$$A \cdot \tilde{p} = B$$

$$A = \sum_{i=1}^{n} \frac{A_i}{\left\| \overline{o_i p_{pi}} \right\|} \quad B = \sum_{i=1}^{n} \frac{B_i}{\left\| \overline{o_i p_{pi}} \right\|} \tag{12.4}$$

Los elementos de cada sumando: $A_i, B_i, \left\| \overline{o_i p_{pi}} \right\|$ se calculan exclusivamente con los datos de cada cámara *i*: o_i, p_{pi}. La sencillez del sumatorio final y de la resolución de un solo sistema lineal 3x3 son los puntos fuertes de este método.

En este caso también es posible ajustarse a la relevancia de la información de cada cámara, siempre que se disponga de una medida para ello. Se puede ponderar el sumatorio de la función $\bar{d}(p)$ en la ecuación (12.2), lo que se trasladará a un ponderado igual en los sumatorios de las ecuaciones (12.4) que definen *A* y *B*. Por lo tanto, este añadido no supone ningún problema.

Pese a que el criterio de mínimo error cuadrático medio no equivale al de mínima distancia euclídea media, no es esperable que la diferencia de resultados sea importante en este caso. El mayor problema del MSE es la afección ante outliers, pero si se emplea la estrategia de ponderación de cámaras con información apropiada acerca de la fiabilidad de cada una, este problema debería verse paliado. Por otro lado, la distancia euclídea no es especialmente inmune a outliers, y habría que emplear otra función de error (igual o más complicada de derivar) para eliminar completamente la posibilidad de que un error grave en la localización de una primitiva en una cámara (por, por ejemplo, un falso punto de luz) provocara una desviación palpable en $\tilde{p}$. Esto si no se desea, como antes, calcular todas las distancias relativas entre rectas.

Aunque se han implementado ambos métodos, el sistema final hace uso del primero, por no esperarse el uso de un número de cámaras que justificara el segundo método en términos de eficiencia, y por ser la evolución natural del método implementado para el primer caso probado; el de dos cámaras. Sin embargo, la implementación del segundo método es más fácil de depurar desde cero, al evitar la complejidad del tratamiento de combinaciones de rectas en el primero, y también más elegante matemáticamente, por lo que se ha introducido aquí como alternativa perfectamente válida.

Fase 1 de localización en tiempo real: **Extracción de las primitivas 2d del objeto localizable**

Este submódulo debe extraer las primitivas bidimensionales de cada imagen. Por las razones explicadas en apartados anteriores, no empleará la línea epipolar para la búsqueda de primitivas en todas las cámaras salvo en una, si no que se realizará una búsqueda completa en todas las imágenes. Por tanto, la algoritmia será la misma para todas las cámaras.

Como también se comentó, se emplearán diodos LED como emisores activos, y se reducirá al máximo el tiempo de exposición de la cámara, de modo que la distinción de los puntos será más obvia en términos de luminancia, sin requerir una búsqueda de patrones compleja e inabordable en tiempo real y compartido. Además, la reducción del tiempo de exposición tiene otras consecuencias positivas, como la desaparición de la estela en movimientos rápidos del punto luminoso, inconveniente producido por tiempos de integración demasiado largos. Hay que notar sin embargo que no todas las cámaras de bajo coste admiten una modificación programática del tiempo de exposición, al menos por la interfaz empleada (para detalles sobre esto, ver apartado de implementación), y las que lo admiten, pueden ofrecer valores límite muy dispares. (Aunque se hace referencia a tiempo de exposición, el grado de influencia en el resultado de otros factores como sensibilidad del CCD es transparente desde la interfaz empleada). Los resultados de estas diferencias son que, para unas condiciones de iluminación iguales y típicas, asignando el mínimo tiempo de exposición, con algunos modelos de cámara los objetos de la escena que no poseen iluminación activa quedan oscurecidos de forma matemáticamente irrecuperable (nivel tan bajo que queda enmascarado por el ruido de sensor), y con otros, aunque la bajada de nivel para objetos pasivos es notable, los contornos son recuperables y algunos objetos muy reflectantes pueden alcanzar valores muy altos de luminancia. Por tanto, el algoritmo debe ser flexible ante estas variaciones.

Si se considera un solo punto, la opción más sencilla es fijar un umbral de luminancia en tiempo de compilación, y recorrer cada raster de forma que la primitiva corresponde al último píxel con valor de luminancia mayor que dicho umbral. Si ningún píxel supera el valor, se considera que el punto no está activo o no está dentro del campo de visión de esa cámara en particular.

Esto plantea dos problemas principales: la elección del umbral de forma que sea adecuado para distintos modelos de cámara y distintas condiciones de iluminación, y las falsas alarmas producidas por otras regiones con iluminación activa, o regiones pasivas pero altamente reflectantes.

Dejando la elección del umbral para más adelante, el mayor problema viene de mano de las regiones de iluminación activas, que en un recinto cerrado suelen corresponder a objetos como lámparas y ventanas. Para paliar este problema, se ha optado por restar a cada fotograma de localización en tiempo real una máscara constante y específica para cada cámara, consistente en una media de diversos fotogramas iniciales, posteriormente sometida a una transformación morfológica de dilatación. Se está empleando una "dilatación escalar", generalización de la binaria, y que consiste en sustituir cada punto de la imagen por el máximo en un entorno alrededor de él, dado por el elemento estructurante, como puede verse en la tercera ecuación de (13.1).

$$FOTOGRAMA\ (t) = FOTOGRAMA\ (t) - MASCARA\ (S_n)$$

$$S_n = \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix} \in M_{nxn} \quad / \quad n = 2k+1$$

$$MASCARA\ (S_n) = \frac{\sum_{i=0}^{N} FOTOGRAMA\ (i)}{N} \oplus S_n \tag{13.1}$$

$$I \oplus S_n = J \ / J(x,y) = \max\left(\left\{I(x',y') / x' \in [x-k, x+k], y' \in [y-k, y+k]\right\}\right)$$

Para llevar a cabo esta operación se requiere que durante los primeros instantes (es suficiente un segundo si se trabaja a una razón de 10 fps, de forma que N=10) la escena esté inmóvil, y con todas las fuentes activas de luz funcionando y sin ocluir. La media suaviza fenómenos espurios de fotogramas individuales, especialmente efectos de solapamiendo de frecuencia entre la tasa de captura de la cámara y la frecuencia de funcionamiento de ciertas luces fluorescentes. La dilatación posterior evita el efecto de aureola que se forma alrededor de las regiones de alta luminancia cuando se emplea sólo la media como operador anterior. Este efecto parece deberse a que, por características de funcionamiento de los CCD, en estas regiones fronterizas hay una gran varianza en la luminancia, pero la investigación sobre las causas de este fenómeno no es un objetivo del proyecto. Si se empleara el operador máximo este fenómeno desaparecería, pero también se generarían valores demasiado altos en las regiones de la máscara no relativas a objetos luminosos, por efecto de los outliers en el ruido estático de la imagen. Por lo tanto, es más recomendable emplear el operador media y posteriormente la dilatación, que elegida con un elemento estructurante de dimensiones adecuadas (n=5,k=2 suele ser suficiente) no desperdicia un porcentaje significativo del campo de visión útil de la cámara (hay que tener en cuenta que la máscara también elimina el patrón luminoso a localizar). Además, la dilatación previene también de problemas relacionados con ligeros movimientos físicos de los objetos luminosos interferentes, que pueden ser por causas tan variadas como una lámpara cimbreante en el campo de visión o el cambio de perspectiva del sol en una ventana o un espejo. Cabe decir que frente a esto último (problemas relacionados con el sol y su movimiento) no hay más solución a largo plazo que evitar enfocar este tipo de regiones, no sólo por las falsas alarmas al desactualizarse la máscara calculada incialmente, si no porque incluso con una máscara apropiada, la pérdida de campo de visión útil para la cámara es muy grande.

Para el objetivo opcional de localizar varios puntos simultáneamente puede emplearse información de color. Los motivos para no emplear las rectas epipolares de búsqueda, relacionados con el bajo coste de las cámaras empleadas y la gran cantidad de distorsiones sobre el modelo pinhole, ya se discutieron anteriormente. Para la tarea es especialmente apropiado el uso de un espacio de color YUV, que es una opción de salida configurable programáticamente en la mayoría de las cámaras, y en última instancia fácilmente obtenible a través de una transformación lineal desde RGB, que es

el formato nativo de los sensores CCD. Así, puede emplearse la componente Y de luminancia para lo ya mencionado, y las dos componentes de crominancia U y V para determinar la correspondiencia entre primitivas bidimensionales asociadas a un mismo punto tridimensional en distintas cámaras. En la figura 13.1 puede apreciarse gráficamente la relación entre las componentes U y V y la saturación para una caso típico de resolución de 8 bits (sin signo) por componente.

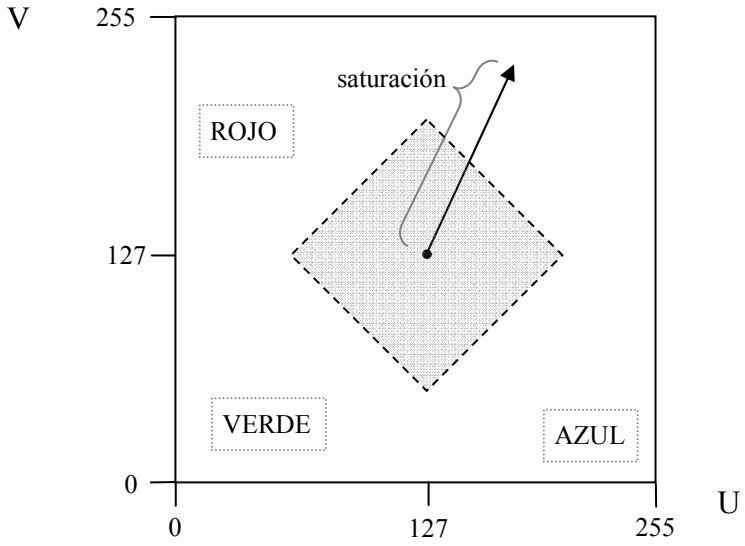


Figura 13.1

Además, se suele dar el hecho de que las fuentes de luz interferentes, si bien con cierta componente de saturación, no suelen ser altamente luminosas y altamente saturadas al mismo tiempo, ya que salvo en casos específicos, se suele buscar con ellas una iluminación neutra de una estancia. Esto se puede aprovechar como información adicional para filtrar estas fuentes si los puntos localizables emplean siempre una iluminación activa muy saturada, lo cual se puede conseguir sin pérdida de rendimiento energético con el ya mencionado uso de LED's. Para esto, se puede emplear un criterio basado en norma-0 (empleando valores absolutos), muy ligera computacionalmente. El umbral, nuevamente, debe seleccionarse apropiadamente. Para uno arbitrario, la frontera resultante viene representada en la figura 13.1 con trazo discontinuo, quedando sombreada la zona excluída.

La proposición (13.2) resume lo expuesto, nuevamente considerando una profundidad de 8 bits para todas las bandas:

$$\begin{aligned}&((X,Y) = \text{PRIMITIVA}) \Rightarrow \\ &\quad (\text{IMAGEN}(X,Y).Y - \text{MASCARA}(X,Y) < \mathit{ThrLuma}) \\ &\quad \mathit{AND} \\ &\quad (|\text{IMAGEN}(X,Y).U - 128| + |\text{IMAGEN}(X,Y).V - 128| < \mathit{ThrChroma})\end{aligned} \tag{13.2}$$

Nótese que la implicación no es doble, pues puede haber, y de hecho es normal que haya, más puntos en la imagen que número máximo de primitivas esperado. Algunos podrán ser falsas alarmas, y muchos de ellos formarán clústeres cercanos por corresponder al mismo emisor. Una forma sencilla de lidiar con estos dos fenómenos es, para cada primitiva esperada asignar unas coordenadas (U,V) ideales, y quedarse con el candidato del conjunto descrito por la proposición 13.2 que mantenga una distancia mínima con él en el plano U,V. Esta es la estrategia que se ha seguido en la implementación.

Queda por discutir la elección apropiada del umbral de luminancia, problema al que ahora se suma el análogo del umbral de crominancia. En la implementación del sistema se han empleado valores iguales para todas las cámaras, y de valor constante, aunque se pueden cambiar manualmente en tiempo de ejecución. Se ha comprobado que, para profundidad de 8 bits/banda, condiciones típicas, y el uso de la máscara antes mencionada, valores cercanos a 50 tanto para *ThrLuma* como para *ThrChroma* suelen dar los mejores resultados.

Otra opción sería establecer el valor de *ThrLuma* cuando se calcula la máscara y en función de los valores de esta, ya que la necesidad de mantener la misma configuración de hardware y las mismas condiciones de iluminación se da para ambas variables (o sea, sus periodos de validez coinciden), y la mayor o menor intensidad de la máscara está en cierta medida relacionada con la mayor o menor intensidad de las posibles fuentes de interferencia que *ThrLuma* debe superar. De hecho, las prácticas con la implementación muestran que prácticamente la mayoría de las interferencias provienen de la reflexión de la luz emitida por las fuentes primarias, ya filtradas por la máscara, sobre objetos con bastante color (ropa del usuario, soporte físico de los LED's), lo que provoca nuevas fuentes muy saturadas y aún de suficiente brillo. Sin embargo, se ha implementado alternativamente esta estrategia, empleando un factor de proporcionalidad constante entre el *ThrLuma* de cada cámara y la media de los valores de su máscara, y no ha ofrecido resultados mejores a la de los umbrales manuales cuidadosamente seleccionados.

Otra posible mejora sería emplear valores variables de *ThrLuma* y *ThrChroma* , que se determinaran de forma adaptativa durante el periodo de localización en tiempo real. Por ejemplo, podrían determinarse sus valores en función de los falsos negativos o los falsos positivos detectados en cada iteración del sistema. Sin embargo, la tasa de falsos negativos no podría determinarse, al no disponer el sistema de conocimiento a priori sobre la cantidad de puntos localizables dentro del campo de visión de una cámara. Una tasa de falsos positivos podría obtenerse registrando en cada fotograma cuántos puntos cumplen la proposición (13.2), y aumentando los umbrales (haciéndolos más selectivos) según aumenta esta cantidad. Sin embargo, dado que el algoritmo píxel-a-píxel, como precio de su agilidad, sólo entiende de píxeles y no de qué conjuntos conexos de estos constituyen regiones pertenecientes a una misma fuente, esta estrategia podría verse indeseablemente afectada por la cercanía a la cámara de una fuente luminosa a localizar. Concretamente, la localización de un punto con una crominancia determinada podría sufrir las consecuencias de un aumento de los umbrales debido a la aproximación a la cámara de otro punto con otra crominancia. Además, el hecho de que existan falsos positivos aún con umbral de crominancia constante pero seleccionando el punto que más cercano queda a alguna de las coordenadas (U,V) ideales, indica que en lo que a

ThrChroma respecta, aumentar el umbral hasta obtener un único candidato no solucionaría los problemas existentes. Con todo esto, parece que la mejor alternativa sería aplicar un algoritmo adaptativo para determinar las propias coordenadas (U,V) ideales, en función de la crominancia la última serie de puntos detectados que se tuviera constancia de que son acertados. Esto podría obtenerse mediante una comunicación bidireccional con la fase 2 de localización, que en función de la incoherencia para dos puntos detectados en dos cámaras distintas para la misma coordenada (U,V) ideal[5], podría determinar con bastante confiabilidad (aunque no absoluta) si la detección de ambos puntos ha sido acertada. Esto se puede generalizar para más de dos puntos. Sin embargo, esta comunicación bidireccional complica bastante el modelo propuesto, donde se persigue la independencia de cada fase del sistema con respecto a sus posteriores.

Precisamente en esta misma línea, podría seguirse una estrategia de laxitud en los umbrales de la fase 1, relegando a la fase 2 la tarea de discernir cual de los múltiples candidatos a cada coordenada (U,V) ideal en cada cámara es el adecuado, en función de su incoherencia con los otros candidatos para la misma coordenada (U,V) ideal en las otras cámaras. Sin embargo, se ha procurado evitar esta estrategia de "delegación de responsabilidad" (entre todas las fases del sistema completo, no sólo entre estas) ya que si se toma como un hábito, puede caerse en el error de no aprovechar al máximo la información disponible para cada fase para rebajar la probabilidad de error de su salida, delegando siempre en la siguiente. Esto provocaría un aumento del peso computacional del conjunto al ser necesaria una evaluación de múltiples combinaciones de salidas de cada fase en la siguiente. Sin embargo, quedaría como una posible mejora el desarrollar comunicación bidireccional o delegación de responsabilidad entre las fases implementadas, si se hace cuidadosamente.

La detección basada en umbrales es muy básica, y falla con facilidad si objetos reflectantes se interponen en el camino de una fuente de luz potente y directa, como una ventana. Si se dispusiera de hardware muy potente y especializado, podrían aplicarse métodos más pesados computacionalmente, basados en convoluciones con máscaras. Sin embargo, con la necesidad de que el sistema completo funcione en tiempo real en el hardware de propósito general disponible, que además deberá ser compartido con la aplicación que haga uso de la localización del patrón, estas técnicas se vuelven prohibitivas.

[5] Podrían determinarse coordenadas (U,V) ideales distintas para cada cámara, recomendable si son cámaras distintas que pueden percibir un mismo patrón con distinta crominancia, pero esto complicaría más el proceso.

3.3. CARACTERÍSTICAS DE LA IMPLEMENTACIÓN

SOFTWARE:

La implementación software se ha realizado sobre los lenguajes Matlab R2008b y C [6]. Matlab (a veces denominado "Lenguaje M" para distinguirlo del entorno de desarrollo), por estar especialmente enfocado al manejo de matrices, es muy conveniente para la gran cantidad de operaciones de álgebra lineal que ha de realizar el sistema. Su facilidad para manejar matrices multidimensionales, ya como tablas en general y no sólo desde el punto de vista algebraico, es también útil para las múltiples ocasiones en las que el sistema debe manejar estructuras de datos con productos cartesianos de varios índices. Por ejemplo, sobre un array tridimensional con las distintas coordenadas de las proyecciones de distintos puntos en distintas cámaras, existen rutinas muy eficientes para el acceso a subarrays de una y dos dimensiones, que en otros lenguajes deberían ser importadas desde librerías externas o implementadas a bajo nivel.

Además, Matlab incluye una enorme colección de *toolboxes*, de entre las cuales se han empleado dos:

- La *Image acquisition toolbox,* como interfaz de acceso a los búferes de imágenes de las cámaras, y como interfaz para configurar programáticamente parámetros como el tiempo de exposición.
- La *Image processing toolbox,* con ciertas funciones de procesado de imágenes, como la de proyección sobre una recta de ángulo arbitrario, que se ha empleado para la construcción de la Transformada de Hough, el etiquetado de regiones conexas, o las operaciones morfológicas como erosión y dilatación, que se han empleado tanto en su versión binaria como escalar.

En general, Matlab trata a las imágenes como matrices, pero a diferencia de otros lenguajes permite operaciones matriz-escalar como la comparación, lo que agiliza enormemente tareas como la umbralización, al poderse prescindir de bucles anidados.

Sin embargo, por los mismos motivos, para realizar con eficiencia los cálculos requiere que éstos se expresen en formato vectorial, evitando el empleo de bucles. Para cierto tipo de operaciones esto lleva a un código especialmente ofuscado, y en ocasiones es teóricamente imposible vectorizar un algoritmo, no quedando más remedio que emplear un bucle, produciéndose así pérdidas de rendimiento que en otros lenguajes imperativos con un control de flujo tradicional no se darían. Además, Matlab es un lenguaje interpretado, lo que añade una sobrecarga de operaciones al tener que *parsear* el código en tiempo de ejecución (existen técnicas avanzadas para minimizar la pérdida de rendimiento, pero aún así en general un lenguaje interpretado tiende a ser menos eficiente que uno compilado). Su paradigma de alto nivel, cercano a las estructuras matemáticas y alejado de la arquitectura de la máquina, impide que ciertas operaciones se puedan resolver con la eficiencia que permiten de otros lenguajes. El entorno de

[6] Para una indroducción a cada uno de los dos lenguajes, y también como referencia, puede consultarse respectivamente:
Gil Rodríguez, M. *Introducción a Matlab y Simulink para ciencia e ingeniería.* 2003. ISBN: 8479785969.
Kernighan, B W.; Ritchie, D. M. *The C programming language*. Prentice Hall, 1988. ISBN: 0131103628.

desarrollo de Matlab también incluye un compilador, que sirve tanto para crear autoejecutables como para crear librerías precompiladas a partir de código Matlab. Sin embargo, las pruebas realizadas con un bucle recorriendo una imagen con ruido blanco uniformemente distribuido, y realizando operaciones per-píxel, muestran que aunque se consigue mayor rendimiento que con el mismo código interpretado, este aún es menor que en el caso del mismo algoritmo implementado sobre C. La diferencia tanto hacia un caso como hacia otro es de aproximadamente un orden de magnitud, aunque estos resultados pueden depender de la arquitectura de la máquina, los compiladores empleados, el estado de la cola de procesos del SO, etc, por lo que se han obtenido con un propósito meramente orientativo.

Por los motivos expuestos, se ha elegido implementar en C la rutina que recorre los raster que devuelven las cámaras para localizar las primitivas bidimensionales, que es el núcleo de la fase 1 de localización. Se ha empleado el compilador C incorporado en el entorno Matlab, además de una herramienta que proporciona la capacidad de envolver el código objeto producido, de forma que pueda ser invocado desde el entorno de Matlab. Esto se consigue a través de una función intermedia `mexFunction()` escrita en C (opcionalmente en el mismo archivo fuente) y que actúa de interfaz entre las funciones con la lógica deseada y las estructuras nativas de Matlab. Para la comunicación superior se vale de llamadas a funciones incluídas en una librería `mex.h` proporcionada por el entorno, que devuelven punteros a tipos básicos de C (generalmente *double*, por ser su equivalente en Matlab el tipo por defecto). Estos punteros se pueden indireccionar directamente o con un offset variable según la variable original en matlab fuese un escalar o un vector/matriz (como es el caso de las imágenes). El resultado tras la compilación es un archivo .mex (Matlab-EXecutable) que ha de situarse junto con los ficheros M correspondientes. Se han procurado seguir técnicas de optimización C básicas, como recorrer el array bidimensional siguiendo orden de almacenamiento en memoria.

Por otro lado, en ciertos submódulos, aunque se ha usado Matlab para su implementación, se ha evitado deliberadamente el uso de funciones de muy alto nivel disponibles en las librerías de dicho lenguaje. Un ejemplo de ello es la resolución del sistema de ecuaciones no lineales, para lo cual Matlab ofrece ciertas funciones de uso bastante general, como `fsolve()`, pero cuyo uso se ha sustituido por una implementación manual de un método específico. Los motivos de esta elección son un mayor control sobre el método, la posibilidad de adaptar este a las condiciones específicas del problema, y una mayor portabilidad para una hipotética migración del código del sistema completo a un lenguaje de más bajo nivel, como C/C++. Este último caso tendría su justificación tanto en términos de eficiencia como de sencillez de uso para el usuario final (el código M requiere la instalación de Matlab en el sistema operativo huésped, o como mínimo la instalación de ciertas librerías). En concreto, la migración completa del sistema de localización a un lenguaje más eficiente podría ser especialmente recomendable, por las necesidades del tiempo real. Es probable que para ello hubiera que emplear C++ o C#, pues muchas librerías que permiten el acceso a flujos de vídeo están desarrolladas para estos lenguajes.

Los archivos .m (Matlab) y .mex (C) se han estructurado en una jerarquía de directorios semejante a la empleada en la descripción teórica del sistema.

/
shell.m
- CamRetriever
  - *CamRetriever.m*
- CamCalibrator
  - Find2dPatt
    - *Find2dPatt.m*
    - *ProcessSnapshot.m*
    - *SearchCalibPattern.m*
    - *SearchCalibPattLocal.m*
  - Solve3dPatt3ecs
    - *Solve3dPatt3.m*
    - *Solve3dPattOneCam3.m*
    - *NewtonRaphson3.m*
    - *Funcion3.m*
    - *JacobianoFuncion3.m*
  - Solve3dPatt5ecs
    - *Solve3dPatt5.m*
    - *Solve3dPattOneCam5.m*
    - *NewtonRaphson5.m*
    - *Funcion5.m*
    - *JacobianoFuncion5.m*
  - SolveRelativePos
    - *SolveRelativePos.m*
- ObjLocator
  - *InitLocator.m*
  - *PaintStaticScene.m*
  - *ProcessInstantScene.m*
  - *ProcessFrame.m*
  - *findPoints.mex*
  - *Locate3d.m*

Se omitirán detalles más profundos sobre la implementación. Sin embargo, con vistas al posible uso del sistema por parte de una aplicación usuaria, es la función `Locate3d()`, ubicada en *Locate3d.m* , la que devuelve los n puntos localizados en el espacio en una matriz Matlab de n filas y 3 columnas.

A pesar del Objetivo 4, en la implementación no se ha construido Interfaz Gráfica de Usuario, sino que el control se realizará por consola de comandos. Por el principio de separación entre lógica de negicio y de presentación, la interfaz gráfica podría ser construida de forma independiente. Para una descripción de la consola de comandos orientada a usuario, se puede consultar el apéndice “Instrucciones de usuario del sistema implementado”.

HARDWARE:

Las pruebas de ejecución se han realizado con dos cámaras Logitech, modelo S7500, y una adicional modelo QuickCam Pro 4000. En ambos casos se trata de cámaras *off the shelf*. Las características técnicas como la distancia focal equivalente o la resolución nativa del sensor no son publicadas por el fabricante, por lo que no se ha dispuesto de su conocimiento a priori.

Los resultados de carga computacional expresados en tiempo se han obtenido en un ordenador con procesador Intel T5450@1.66Ghz, con el Sistema Operativo Windows XP Profesional SP3.

PATRÓN LOCALIZABLE

Se han probado varias alternativas para conseguir emisores de luz puntuales, omnidireccionales, de bajo coste y bajo consumo, esto último para conseguir una buena autonomía en un mando con batería. Desde el primer momento se ha optado por emplear LED's como transductores, por sus características de elevada eficiencia energética, durabilidad, y recientemente, luminosidad. Se pueden encontrar modelos de 33 lumen de luminosidad, temperatura de color 3300K, y consumiendo 1,2W de potencia eléctrica. Además, dado que el espectro de emisión está concentrado alrededor de una longitud de onda específica en función de los elementos químicos involucrados en la unión PN, se pueden emplear emisores que provoquen huellas de crominancia distintas sin recurrir a filtros basados en la absorción de las longitudes de onda no deseadas, que implicarían una pérdida grande de eficiencia.

Sin embargo, la mayoría de los LEDs son marcadamente direccionales, lo cual supone un inconveniente para los propósitos del sistema, que ha de localizar la fuente independientemente de su orientación. Las causas de su direccionalidad son dos: por un lado, la unión PN emite con un patrón semiesférico por estar montada sobre una superficie que impide el paso de la luz. En muchos casos la superficie es ligeramente cónica o parabólica, de modo que se consigue un patrón de direccionalidad con simetría axial y apertura del orden de 160º. Por otro lado, la mayoría de los encapsulados plásticos actúan a modo de lente para agudizar más la direccionalidad, ya que en la mayoría de aplicaciones esta resulta beneficiosa. Las aperturas que provocan la mayoría de los encapsulados rondan los 100-120º.

Para solucionar este problema, se ha probado a emplear una cápsula esférica de plástico translúcido envolviendo un LED, de forma que provoque por dispersión un diagrama de direccionalidad más homogéneo, pero esto requiere de LED's de alta potencia eléctrica (más de 1W) para superar la pérdida de brillo puntual provocada por la absorción del material y el aumento de su superficie. Se ha optado entonces por emplear LED's con encapsulamiento propiamente translúcido, que aunque siguen adoleciendo de una cierta pérdida de rendimiento energético, mantienen una superficie pequeña, y por tanto un brillo más alto para igual luminosidad. El fabricante ofrece como dato una apertura de 120º. Las características eléctricas y lumínicas de cada modelo empleado son:

Longitud de onda: 630nm (rojo) Tensión: 2,1V Intensidad: 80mA typ. Luminosidad: 5 Lumen max	Longitud de onda: 470nm (azul) Tensión: 3,4V Intensidad: 80mA typ. Luminosidad: 2 Lumen max	Longitud de onda: 520nm(verde) Tensión: 3,4V Intensidad: 80mA typ. Luminosidad: 8 Lumen max

Sin embargo, la directividad sigue siendo notable, como puede apreciarse en los puntos de la izquierda en la figura (14.1). En coherencia con lo dicho antes, la luminancia percibida decrece a partir de los 90º con respecto al eje axial del LED, de forma que aunque aún es claramente visible, en presencia de interferencias, su valor máximo es más susceptible de ser superado por ellas. Hay que recordar que el algoritmo empleado es muy simple porque se requiere aligerar al máximo el funcionamiento de la fase 1 de localización en tiempo real.

Existe la sencilla solución de aumentar la potencia de los LEDs hasta que las direcciones menos favorecidas alcancen una intensidad suficiente para saturar el sensor en esas mismas condiciones. Sin embargo, se descarta por lo incómodo e incluso dañino que resultaría para el usuario (a más de 6,7mA comienza a ser molesto), y por el gasto energético excesivo que supondría.

Se ha optado por modificar el encapsulado de los LEDs, de forma que se sustituya el efecto lente de la cabeza, que concentra hacia el eje axial los rayos más desviados con respecto al mismo, por un efecto espejo que se produce quedar el ángulo de incidencia por encima del límite de reflexión total impuesto por la ley de Snell. Así, se envían parte de esos rayos hacia direcciones con más de 90º con respecto al eje axial, y se consigue una distribución algo más homogénea, como puede apreciarse en los puntos de la derecha de la figura (14.1). En todo caso, es una solución artesanal, y su objetivo es mostrar que un diseño específico para la geometría del encapsulado podría reducir considerablemente la direccionalidad del LED y mejorar por tanto los resultados de localización.

LED’s empleados.
Modelos de 630nm

Ángulo: 0 grados.
Luminancia máxima respectivamente: 235,235.
Intensidad: 6,7 mA
Distancia Cámara-LED’s : 1.2m

En esa situación, ambos LEDs saturan el sensor. Por características de la cámara, el valor de saturación es 235 y no 255.

Ángulo: 45 grados.
Luminancia máxima respectivamente: 235,224.
Intensidad: 6,7 mA
Distancia Cámara-LED’s : 1.3m

Se puede apreciar el efecto de la luz trasera contra la placa de prototipos en el LED modificado.

Ángulo 135 grados.
Luminancia máxima respectivamente: 191,219.
Intensidad: 6,7 mA
Distancia Cámara-LED’s : 1.3m

El encapsulado original ofrece problemas para ángulos en este entorno.El encapsulado modificado mantiene una huella más constante.

Izquierda: Encapsulado original

Derecha: Encapsulado modificado

Figura 14.1

4. CONCLUSIONES Y POSIBLES MEJORAS

Tanto a lo largo del planteamiento inicial y del desarrollo teórico, como durante la implementación del sistema, así como a la vista de los resultados experimentales, se han ido anotando varias posibles modificaciones sobre el sistema.

Algunas son alternativas de ámbito local, consistentes en la adición o sustitución de un determinado submódulo dentro del sistema completo. Muchas de estas ya se han descrito en sus apartados correspondientes, justificando la elección de otra opción pero manteniendo la posibilidad de una alternativa, con sus ventajas y sus inconvenientes. Ejemplos de esto son el empleo de un método de búsqueda por gradiente en la fase 2 de calibrado, o el método alternativo expuesto en la fase 2 de localización. También el uso de puntos de referencia (U,V) ideales adaptativos en la fase 1 de localización, o la delegación de responsabilidad de la fase 1 a la fase 2 de localización.

Igualmente de ámbito local, hay posibles mejoras que no se han mencionado, pero cuya implementación, si bien más compleja, mejoraría ciertos aspectos del sistema. Algunos ejemplos son:

- Si se localizan varios puntos a en el espacio, y entre ellos se mantienen ciertas relaciones de distancia conocidas a priori, esto podría emplearse como información redundante en la etapa 2 de localización, que ayudaría a corregir errores. Un ejemplo sería que dos puntos se mantuvieran a una distancia constante por depender de un mismo soporte físico, como un mando cilíndrico que soportara un LED en cada extremo. Sin embargo, en este caso, la redundancia introducida no sería lo suficientemente fuerte como para solventar el problema más importante: que empleando dos cámaras, la imagen de un punto esté ocluida en una de ellas.

- La oclusión también plantea problemas incluso si se emplean múltiples cámaras y para un punto determinado sólo se ocluye una de ellas en un instante dado. Al emplear un operador estadístico que asigna peso a la información proporcionada por todas las cámaras, para así minimizar ciertos errores asociados a cada una, se obtiene un efecto perjudicial de "saltos discontinuos" cada vez que una cámara pierde o gana contacto visual con el punto a localizar. Estos saltos se deben a que la cámara introduce o deja de introducir ciertas componentes constantes que existen en sus funciones de error, debidas a características inherentes a su óptica, o a imprecisiones de cualquier tipo en su proceso de calibrado. La solución de este problema podría pasar por emplear un método que pudiera modificar dinámicamente el peso asociado a la información proporcionada por cada cámara (ésto se discutió en el apartado correspondiente a esta fase), e introducir un suavizado artificial en la pérdida o recuperación de la información asociada a una cámara. Este suavizado crearía una situación de continuidad entre el estado previo al evento y el posterior, prolongando, por tratarse de un sistema causal, el previo sobre el posterior. Sin embargo, la implementación de esta técnica no es trivial, ya que hace falta no solamente tener la capacidad de asignar pesos dinámicamente, sino también de recrear el sesgo de la fuente de información perdida mientras la posición del punto cambia, que es lo que puede estar sucediendo cuando se ocluye la vista para una

cámara. Además, para que la percepción de la transición de *n* a *n-1* fuentes fuese menor por parte del usuario, sería conveniente relegar la operación de reducción progresiva del peso de la información de la fuente perdida a una situación de movimiento para el punto. Si se hace estando el punto quieto, el movimiento producido por la operación descrita sería muy visible para el usuario, al contrastar con la situación estática del patrón real.

Otras conclusiones son de carácter más global, y atañen a decisiones iniciales que afectan a gran parte del proyecto. Se pueden destacar dos.

- Con respecto a la implementación, se comentó ya la mejora en eficiencia que conllevaría haber empleado un lenguaje de más bajo nivel y compilado para implementar el módulo de localización. A la vista de las grandes diferencias de rendimiento entre Matlab y C, para la labor en tiempo real que lleva a cabo este módulo resultaría especialmente recomendable, permitiendo trabajar con resoluciones mayores o con un número mayor de cámaras. Además, se ha comprobado que aunque la plataforma Matlab ofrece la posibilidad de compilar el código, la eficiencia del resultado sigue siendo en general menor que la del módulo que realiza la misma tarea codificada en C.

- Por otro lado, en vista del esfuerzo invertido en el diseño e implementación del sistema desde cero, y con los resultados obtenidos en la mano, la decisión de no emplear bibliotecas puede ser valorada críticamente. Si bien como ya se expuso, los objetivos del sistema no requieren grandes niveles de precisión, parece claro que implementando un método de calibrado que incluyera distorsiones ópticas en el modelo, como el Tsai, o haciendo uso de bibliotecas para calibración de cámaras ya implementadas, como las mencionadas en el apartado correspondiente, se habría conseguido más precisión en un tiempo de desarrollo semejante. En lo que respecta al módulo de localización, por ser matemáticamente más sencillo, la decisión de implementarlo desde cero parece más acertada, aunque en el caso de haber empleado un modelo de cámara con modelado de distorsiones, habría sido modificado para hacer uso de esa información. En este sentido, por ejemplo, si el modelado de distorsiones radiales sobre las cámaras de bajo coste hubiese reducido suficientemente la incoherencia en la etapa de localización (de lo cual no se puede tener seguridad con los resultados aquí obtenidos, porque estas cámaras introducen muchas más distorsiones), habría sido posible el uso eficiente de rectas epipolares de búsqueda. Esto tendría consecuencias importantes en la eficiencia del sistema, además de permitir usar emisores de luz no diferenciados.

  Sin embargo, hay que señalar la característica propia al método desarrollado para el sistema, y es que necesita únicamente un patrón de calibrado de 4 puntos (3 si se conoce la distancia focal). Al contrario, los métodos consultados (Duda y Hart, Tsai, el implementado en openCV) requieren de un número muy alto de puntos, en el caso de Duda y Hart sin modelar distorsiones ópticas. En cuanto a la coplanariedad, es opcional tanto en el método desarrollado como en otros como el Tsai, dando lugar a pequeñas modificaciones. Si bien con cualquier impresora es sencillo disponer de un patrón con muchos puntos coplanares, la eficiencia (en términos de cantidad de información) del método desarrollado e implementado para este sistema puede ser útil en situaciones donde no se controlara el entorno y un sistema de visión artificial tuviera que hacer uso de información muy escasa. En concreto, el método puede

usarse no sólo para calibrado de un conjunto de cámaras que se usen posteriormente para visión estereoscópica, sino también para la tarea de localizar una cámara, y por ende su soporte, en un entorno arbitrario. Esta tarea puede ser un objetivo por sí misma en aplicaciones, por ejemplo, de robótica. De la misma forma, puede emplearse para localizar un solo patrón compuesto por múltiples puntos con distancias mutuas conocidas, empleando una sóla cámara. Incluso existen motivos para suponer que, con las modificaciones adecuadas, esta tarea podría llevarse a cabo en tiempo real. Esta posibilidad se discutirá con un poco más de detalle un poco más adelante, como último punto del apartado de conclusiones y mejoras.

Por otro lado, si se deseara añadir el modelado de distorsión radial al sistema, se podría hacer sobre el diseño actual, sin necesidad de hacer uso de un método como el Tsai. Para ello, podría añadirse una etapa previa a la de calibrado existente, donde se empleara la imagen de lo que se tuviera certeza que es una línea recta en entorno, o un conjunto elevado de puntos pertenecientes a ella. Mediante una búsqueda por gradiente, podría resolverse el coeficiente k que minimizara el ECM de la regresión lineal de los puntos sometidos a la transformación inversa a la expresada en la ecuación (2.2). Podría emplearse el modelo de distorsión más general expuesto en la ecuación (2.1), pero empleando métodos de búsqueda por gradiente más complejos. Esto anularía lo antes expuesto sobre el bajo número de puntos necesario para el calibrado, pero al tratarse de modelar las distorsiones, es una necesidad matemática insalvable, y en cualquier caso, al tratarse de una etapa previa totalmente diferenciada, podría emplearse u obviarse en función de la información disponible en el entorno. No es un objetivo aquí analizar el método de Tsai, por lo que no se entra en comparaciones de eficiencia y estabilidad numérica entre ambos métodos. Sin embargo, como característica en común, se puede recordar que el método de Tsai también hace uso de dos etapas diferenciadas en las que se resuelven conjuntos distintos de parámetros.

Como último punto, se puede entrar en algo que ya ha sido señalado poco antes, y que se podría considerar como alternativa dentro del diseño del sistema (y potencialmente una mejora), aunque implicaría cambios muy grandes que repercutirían incluso sobre el Objetivo 1 marcado inicialmente (emplear visión estereoscópica). La fase 2 del proceso de calibrado de una cámara, que necesita de un patrón con 3 puntos si se conoce la distancia focal, podría llegar a funcionar en tiempo real, suponiendo que se dispone de una fase 1 que también pudiera hacerlo simultáneamente. En este proyecto no se ha investigado en esta dirección pues el proceso de calibrado se va a ejecutar de forma aislada y basta con que no se demore excesivamente. Pero de funcionar en tiempo real, podría emplearse el propio patrón de calibrado como patrón localizado en tiempo real y actuar como apuntador, para lo cual sólo sería necesaria una cámara. Las ventajas de esto podrían compensar la desventaja de emplear un método de búsqueda no lineal en tiempo real, y de necesitar un patrón localizable más complejo. La posibilidad de uso en tiempo real se basa en las siguientes razones:

- Variando las condiciones de parada de la fase 2 se puede conseguir un aumento considerable de velocidad, renunciando a cierta precisión.
- El método de 3 puntos es lógicamente más ligero, y dado que la distancia focal se asume constante para una cámara, sería suficiente con ejecutar el de 4 sólo una

primera vez. Por tanto, el patrón debería ofrecer tres puntos no alineados, y directamente se conocería no sólo su posición sino su orientación.

- La similitud en la posición entre distintas iteraciones podría aligerar hasta varios órdenes de magnitud el tiempo medio de resolución del sistema no lineal, pues la solución de la anterior podría ser, para la mayoría de los casos, un buen punto de partida para el método de Newton-Raphson. Y aun en los casos en los que no fuera así, debido a un movimiento rápido del patrón o a una mala orientación, podría solucionarse con una batería de búsqueda alrededor de dicho punto, más pequeña que la genérica que se usa en el sistema implementado.
- Se asume una fase 1 capaz de entregar las primitivas de los 3 puntos al mismo tiempo y también en tiempo real, para lo cual los tres puntos localizables del nuevo patrón deberían ofrecer una facilidad de detección similar a la mencionada para el patrón localizable de este proyecto, como consistir en emisores activos de luz bajando el tiempo de exposición de la cámara.

5. PRESUPUESTO

La elaboración del proyecto se ha dividido en tres etapas diferenciadas:

- Estudio teórico previo: Estudio de fuentes y desarrollo de métodos propios:
  120 horas-hombre (dedicación a tiempo parcial)

- Implementación:
  120 horas-hombre (dedicación a tiempo total)

- Redacción de la memoria:
  30 horas-hombre (dedicación a tiempo parcial)

Considerando un coste medio de 10€/hora-hombre, se estima un coste total de trabajo humano del orden de 2700€.

Los costes materiales se detallan a continuación:

MATERIAL	PRECIO UNITARIO (€)	UDS	COSTE TOTAL (€)	PORCENTAJE DE USO **	COSTE AMORTIZADO (€)
Logitech S7500*	40	2	80	36%	29
Logitech QuickCam Pro*	30	1	30	36%	11
LED 10mm 2 lumen	3	6	18	1%	1
Placas de prototipos	5	2	10	10%	1
Equipo informático*	500	1	500	36%	180
Licencia individual Sistema Operativo	150	1	150	36%	54
Licencia individual Matlab Suite	1000	1	1000	36%	360
Licencia individual Image Processing Toolbox	200	1	200	36%	72
Licencia Image Acquisition Toolbox	200	1	200	36%	72
Licencia Matlab Compiler	500	1	500	36%	180
COSTE MATERIAL TOTAL					**960**

[*] Artículo no nuevo.
[**] Para material informático (hardware y software) se considera desvalorización lineal en 5 años sin valor residual. Con periodo de uso aproximadamente igual a 1 año, porcentaje de uso igual a 36%.

COSTE DE TRABAJO HUMANO	2700€
COSTE MATERIAL	960€
COSTE TOTAL DEL PROYECTO	**3660€**

BIBLIOGRAFÍA

Visión máquina:

DE LA ESCALERA HUESO, Arturo de la. *Visión por computador : fundamentos y métodos.* Prentice Hall, 2001. 274 p. ISBN: 9788420530987.

DUDA, Richard O.; HART, Peter E. *Pattern classification and scene analysis*. John Wiley & Sons, 1973. 482 p. ISBN: 0471223611.

GONZÁLEZ JIMÉNEZ, Javier. *Visión por computador.* Paraninfo, 2000. 429 p. ISBN: 8428326034.

TSAI, Roger Y. *A versatile camera calibration for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses*. IEEE Journal of Robotics and Automation, VOL. RA-3, NO. 4, August 1987.

Métodos numéricos:

BURDEN, Richard L.; DOUGLAS FAIRES, J.. *Análisis numérico.* Palmas, Oscar (trad.). 7ª ed.Thomson, 2002. ISBN: 970-686-134-3.

CHAPRA, Steven C.. *Applied Numerical Methods with Matlab*. 2nd ed. Mc Graw Hill, 2008. ISBN: 978-0-07-313290-7.

INFANTE DEL RÍO, Juan Antonio; REY CABEZAS, José María. *Métodos numéricos*. 2ª ed. Pirámide, 2002. ISBN: 84-368-1724-9.

MATHEWS, John H.; FINK, Kurtis D.. *Métodos numéricos con Matlab*. Paíl Escolano, Pedro José (trad.). 3ª ed. Prentice Hall, 1999. 736 p. ISBN: 84-8322-181-0.

Álgebra general:

MERINO GONZÁLEZ, Luis M.; SANTOS ALÁEZ, Evangelina. *Álgebra lineal con métodos elementales*. 1997. ISBN: 84-605-9431-9.

Implementaciones:

[*Impl.1*] BRADSKY, Gary R. *Learning OpenCV*. O'Reilly, 2008. 555p. ISBN: 9780596516130

[*Impl.2*] http://www.vision.caltech.edu/bouguetj/calib_doc/

[*Impl.3*] http://www.cs.cmu.edu/~rgw/TsaiCode.html

Apéndice: Instrucciones de usuario del sistema implementado

REQUISITOS DEL SISTEMA:

- **Dos o más cámaras**, no necesariamente iguales. Pueden ser desde cámaras web por interfaz USB hasta cámaras analógicas conectadas a una tarjeta de digitalización. Sin embargo, no todas son compatibles, pues se necesita al menos una resolución QVGA (320x240) con formatoYUV, y control programático del tiempo de exposición. Además, una tasa de fotogramas por segundo muy baja puede degradar el rendimiento del sistema. Aunque algunas cámaras de muy bajo coste no cumplen todas estas características, la mayoría de las actuales sí lo hacen. En cualquier caso, salvo en el caso de los fotogramas por segundo, el software se encarga de filtrar automáticamente las cámaras que no cumplen los requisitos.

- CPU con frecuencia de reloj de al menos 1Ghz (orientativo, sin considerar que se comparta con otra aplicación de cálculo intensivo).

- Entorno **Matlab 2008b** instalado.

- **Patrón de calibrado**. Consistente en una hoja de papel DIN-A4 pegada sobre una superficie plana, de color oscuro, que sobresalga unos centímetros por detrás para crear un marco con suficiente constraste.

- **Patrón localizable**. Consistente en uno o varios LED's , colocados sobre un soporte que los mantenga a distancias mutuas constantes, o independientes unos de otros. Los colores posibles en la implementación actual son ROJO, AZUL, VERDE.

Colocar las cámaras orientadas hacia una zona común, a una distancia de alrededor de 2 metros del centro de esta. Si son dos cámaras, se recomienda que la distancia mutua sea similar, para formar un ángulo de alrededor de 60 grados. Si son más, pueden situarse formando los vértices de un polígono más o menos regular alrededor de la zona común. Evitar un *roll* (ángulo alrededor del eje óptico de la cámara) mayor a 20 grados.

Conectar al menos dos cámaras al ordenador, e instalar los drivers correspondientes. Desactivar cualquier software de *face traking* (seguimiento automático de la cámara mediante zoom digital) o cualquier otra característica automática especial. Asegurarse de que ningún otro programa está haciendo uso de las cámaras.

Desactivar cualquier modo de ahorro de energía de la CPU que haga *downstepping* sobre la frecuencia del reloj principal.

Iniciar la función `shell` para iniciar el programa. Esperar el prompt `#` , que indica que la línea de comandos está preparada. A continuación se hará una descripción de los comandos disponibles, junto con sus opciones.

Comandos básicos:

`retr [-vga|-qvga]`

(*RETRieve*) Busca todas las cámaras conectadas al computador que cumplan los requisitos necesarios del programa, y las inicia. Además, asigna a las cámaras una distancia focal por defecto, cercana a 50mm EFL. Opciones:

`-vga`

(Por defecto) Las inicia con resolución 640x480.

`-qvga`

Las inicia con resolución 320x240. Recomendable para reducir la carga del sistema cuando se trata de más de dos cámaras, o de un procesador lento Cuando la línea de comandos no responde durante la localización en tiempo real, o esta funciona a saltos perceptibles, emplear esta opción.

`calfl1 n`

(*CALibrate Focal Length phase 1*) (requiere `retr`) Primera fase de calibrado de la distancia focal de la cámara `n`. Si la distancia focal por defecto asignada a la cámara `n` es suficientemente fiel a la real, pueden saltarse tanto `calfl1 n` como `calfl2 n.` Puede saberse cual es la cámara `n` invocando `camprop , prw ,` u observando el LED indicador que poseen muchas cámaras.

Participación necesaria del usuario: El usuario muestra a todas las cámaras el patrón de calibrado. Una vez introducido el comando, el usuario dispone de unos segundos para colocarse de pie, con el patrón centrado delante de su cuerpo, colocado en vertical, y en una posición que sea visible para la cámara `n`, a aproximadamente medio metro de distancia de esta. Sonarán seis pulsos sonoros, con un periodo de 1 segundo entre ellos. El tercero y el sexto serán más agudos, y coincidirán con el sistema tomando un fotograma de la escena desde cada cámara. Los dos pulsos graves previos a cada pulso agudo sirven de referencia al usuario. Entre ambos fotogramas, separados por tres segundos, debe realizarse una cierta rotación del patrón, de modo que sus cuatro esquinas se mantengan más o menos en el mismo plano, pero roten del orden de 20º con respecto a un eje perpendicular a dicho plano. El resto de la escena debe mantenerse lo más estático posible.

Después de esto, el sistema mostrará los resultados gráficamente, indicando dónde ha localizado las esquinas del patrón en una imagen tomada por la cámara `n`. Si el resultado no es satisfactorio, se deberá repetir el proceso.

`calfl2 n`

(*CALibrate Focal Length phase 1*) (requiere `calfl1 n`) Segunda fase de calibración de la distancia focal de la cámara `n`. Si la distancia focal por defecto asignada a la cámara `n` es suficientemente fiel a la real, pueden saltarse tanto `calfl1 n` como `calfl2 n`. El sistema, en función de las coordenadas de las proyecciones de las esquinas del patrón halladas para cada imagen en `calfl1 n`, empleando como información a priori la distancia real entre dichas esquinas en el espacio tridimensional (dimensiones DIN-A4), halla la distancia focal de la cámara `n`. El proceso es típicamente instantáneo, pero en ciertos casos puede demorarse unos pocos segundos. Si la incoherencia es superior a 5 mm, se recomienda repetir el proceso desde `calfl1 n.`

`calps1`

(*CALibrate PoSition phase 1*) (requiere `retr`). Primera fase del calibrado de la posición relativa de las cámaras.

Participación necesaria del usuario: El procedimiento a seguir es igual al descrito para `calfl1 n`, pero se realiza una sola vez para todas las cámaras, por lo que el patrón debe situarse de forma visible para todas ellas, a una distancia a cada una de alrededor de 2 metros.

El sistema mostrará los resultados gráficamente, indicando dónde ha localizado las esquinas del patrón en la imagen de cada cámara. Si el resultado no es satisfactorio, se deberá repetir el proceso.

`calps2`

(*CALibrate PoSition phase 2*) (requiere `calps1`). Segunda fase del calibrado de la posición relativa de las cámaras. El sistema, en función de las coordenadas de las proyecciones de las esquinas del patrón halladas para cada imagen en `calps1`, empleando como información a priori la distancia real entre dichas esquinas en el espacio tridimensional (dimensiones DIN-A4), halla las coordenadas tridimensionales de las esquinas. El proceso es típicamente instantáneo, pero en ciertos casos puede demorarse unos pocos segundos. Si la incoherencia es superior a 5 mm, se recomienda repetir el proceso desde `calps1 n.`

`calps3`

(*CALibrate PoSition phase 3*) (Requiere `calps2`) Se inicia la tercera y última etapa de calibrado de la posición relativa de las cámaras, donde el sistema, con la información concerniente a la posición tridimensional del patrón de calibrado desde el sistema de referencia de cada cámara, calcula las transformaciones geométricas entre dichos sistemas de referencia, o lo que es lo mismo, las

posiciones y orientaciones relativas entre las cámaras. Si el determinante expresado en pantalla no es cercano a 1 por al menos dos decimales, se recomienda repetir el proceso desde `calps1 n.`

`loc [-none|-scene|-grid]`

(*Localization*) (Requiere `cal3`) Prepara el sistema para iniciar la localización del patrón en movimiento en tiempo real.

`-none`

(Por defecto) El programa realiza la estimación de la posición del patrón de localización, pero no genera ninguna salida visual.

`-scene`

El programa activa un modo de visualización de las coordenadas estimadas para el patrón de localización que consiste en una representación tridimensional básica de la escena y del patrón.

`-grid`

El programa activa un modo de visualización de las coordenadas estimadas para el patrón de localización que consiste en la visualización de una rejilla que simula ser el suelo bajo el patrón localizable.

`st`

(*Start*) (Requiere `loc`) Inicia la localización en tiempo real. Si `loc` o `rf` han sido llamados antes de cualquier otra llamada a `st` , se construirá para cada cámara una máscara que permitirá obviar las fuentes de luz de la escena que no sean el patrón a localizar.

Participación necesaria del usuario: Durante 1 segundo, debe evitarse la oclusión de cualquier fuente de luz que quede dentro del campo de visión de cualquier cámara (lámparas, ventanas, etc), y estas deben exhibir las mismas propiedades que exhibirán durante el funcionamiento en tiempo real. Por tanto, tanto el usuario como el patrón localizable deben situarse fuera del campo de visión.

`pa`

(*Pause*) (Requiere `st`) Detiene temporalmente la localización en tiempo real.

`rfm`

(*ReFresh mask*) (Requiere `pa`). Resetea el sistema para que en la siguiente ejecución de `st` se vuelva a calcular la máscara. Es más rápido que llamar

nuevamente a `loc`, y útil para situaciones donde las fuentes de luz varían inevitablemente, por ejemplo en el caso de la luz del sol con el paso de un periodo de tiempo del orden de media hora.

`save`

Almacena todas las variables de estado en el archivo "savedSession" en el directorio raíz del programa. Es especialmente útil si se considera que la mayoría de funciones no son fail-safe, por lo que un comando con parámetro fuera de rango, o una ejecución de comandos en un orden inadecuado puede provocar un error en tiempo de ejecución que termine la ejecución del programa, perdiéndose las variables de calibrado obtenidas hasta ese momento. Este comando permite evitar ese tedioso inconveniente.

`load`

Recupera las variavles del archivo "savedSession", en el directorio raíz del programa. Sin embargo, el comando `retr` debe volver a ser llamado, ya que la recuperación de las variables internas del programa no implica el restablecimiento de las conexiones con el hardware de video. Si se resolvieron distancias focales con `calfl1` y `calfl2`, `retr` respetará sus valores y no los reescribirá por aquellos asignados por defecto. Por supuesto, cualquier modificación en la posición de las cámaras durante el periodo en el que la información queda guardada de forma persistente invalidará a la misma. También lo hará con cierta probabilidad una desconexión y reconexión de alguna de las cámaras a nivel del Sistema Operativo, ya que esto puede provocar que `retr` las enumere en un orden distinto, resultando incoherente toda la información almacenada para el orden anterior.

`exit`

Salir del sistema. Es importante para cancelar adecuadamente la conexión al hardware de captura de video.

Comandos avanzados:

`camprop`

(*CAMera PROPerties*) (Requiere `retr`). Lista todas las cámaras, con su número identificador y sus propiedades.

`setmain n`

(*SET MAIN camera n*) (Requiere `retr`). Asigna identificador 1 a la cámara con identificador n, y viceversa. Aunque la cámara 1 establece el sistema de referencia principal, en la implementación actual su información no tiene más peso que la de otras.

`delete n`

Borra de la lista la cámara con identificador n. Útil si su calidad de video degrada el rendimiento del sistema, pero `retr` no la ha filtrado.

`prw`

(PRevieW)(Requiere `retr`).Se muestra para cada cámara una ventana con el video capturado.

`cprw`

(Close PRevieW)(Requiere `prw`). Se cierran las ventanas con el video capturado.

Comandos para desarrolladores:

`loc -2d n`

Modo orientado a debugging, que hace que el funcionamiento en tiempo real consista en mostrar la imagen capturada por la cámara `n` con las primitivas 2D detectadas superpuestas.

`db`

(DeBug). Entra en modo consola nativa de Matlab, lo que permite ejecutar código arbitrario e inspeccionar ciertas variables. Sólo disponible para versiones de no compiladas (actual).